PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L 12/64, 12/40, 12/42

A1

(11) International Publication Number: WO 97/24846

(43) International Publication Date: 10 July 1997 (10.07.97

(21) International Application Number: PCT/SE96/01750

(22) International Filing Date: 23 December 1996 (23.12.96)

(30) Priority Data:
9504681-9 28 December 1995 (28.12.95) SE

(71) Applicant *(for all designated States except US)*: DYNARC AB [SE/SE]; Hagavägen 10, S-171 53 Solna (SE).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: BOHM, Christer [SE/SE]; S:t Paulsgatan 28A, S-118 48 Stockholm (SE). LINDGREN, Per [SE/SE]; Maria Prästgårdsgatan 12, S-118 52 Stockholm (SE). RAMFELT, Lars [SE/SE]; Svanekegränd 1, 2tr., S-164 40 Kista (SE). HIDELL, Markus [SE/SE]; Kyrkvägen 38, S-181 42 Lidingö (SE). SJÖDIN, Peter [SE/SE]; Linnégatan 18B, S-753 32 Uppsala (SE).

(74) Agent: L.A. GROTH & CO. KB; P.O. Box 6107, S-102 32 Stockholm (SE).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE HU, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA UG, US, UZ, VN, ARIPO patent (KE, LS, MW, SD, SZ UG), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG)

Published
With international search report.
With amended claims.
In English translation (filed in Swedish).

(54) Title: METHOD AND ARRANGEMENT FOR NETWORK RESOURCE ADMINISTRATION

(57) Abstract

The present invention relates to a method and an arrangement for centralized and distributed management of the capacity in a circuit switched network with a bus or ring structure, especially a network of DTM-type (Dynamic Synchronous Transfer Mode), which networ uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots fo transfer of data, and wherein each data time slot is associated with a token. In the centralized version, a first node, called server node, i allocated tokens corresponding to all data time slots which are flowing in one direction in a bus or a ring. A second node is requesting token corresponding to a certain capacity from the server node, and the server node makes reservations for and transfers tokens correspondin to requested capacity to the other node in event the server node has requested capacity unutilized. The number of server nodes in th distributed version is between two and the total number of nodes in the bus or ring.

Best Available Copy

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AM	Armenia	GB	United Kingdom	MW	Malawi
AT	Austria	GE	Georgia	MX	Mexico
AU	Australia	GN	Guinea	NE	Niger
BB	Barbados	GR	Greece	NL	Netherlands
BE	Belgium	HU	Hungary	NO	Norway
BF	Burkina Faso	IE	Ireland	NZ	New Zealand
BG	Bulgaria	IT	Italy	PL	Poland
BJ	Benin	JP	Japan	PT	Portugal
BR	Brazil	KE	Kenya	RO	Romania
BY	Belarus	KG	Kyrgystan	RU	Russian Federation
CA	Canada	KP	Democratic People's Republic of Korea	SD	Sudan
CF	Central African Republic			SE	Sweden
CG	Congo	KR	Republic of Korea	SG	Singapore
CH	Switzerland	KZ	Kazakhstan	SI	Slovenia
CI	Côte d'Ivoire	LI	Liechtenstein	SK	Slovakia
CM	Cameroon	LK	Sri Lanka	SN	Senegal
CN	China	LR	Liberia	SZ	Swaziland
CS	Czechoslovakia	LT	Lithuania	TD	Chad
CZ	Czech Republic	LU	Luxembourg	TG	Togo
DE	Germany	LV	Latvia	TJ	Tajikistan
DK	Denmark	MC	Monaco	TT	Trinidad and Tobago
EE	Estonia	MD	Republic of Moldova	UA	Ukraine
ES	Spain	MG	Madagascar	UG	Uganda
FI	Finland	ML	Mali	US	United States of America
FR	France	MN	Mongolia	UZ	Uzbekistan
GA	Gabon	MR	Mauritania	VN	Viet Nam

METHOD AND ARRANGEMENT FOR NETWORK RESOURSE ADMINISTRATION

Description

A. A fast circuit-switching for the next generation of high performance network

Abstract

DTM, Dynamic synchronous Transfer Mode, is a broadband network architecture based on fast circuit-switching augmented with dynamic reallocation of resources. It provides a service based on multicast, multi-rate channels with short set-up delay, and supports applications with real-time requirements on quality of service as well as applications with bursty, asynchronous traffic. This paper describes the DTM architecture and its distributed resource management scheme. Performance analysis results from network simulations are presented. The analysis is performed with respect to throughput and access delay for two network topologies: a dual bus and a grid of dual buses. The effects of varying user requirements, inter-node distance and transfer size are studied for uniform traffic patterns. The results indicate that the overhead for establishing channels is low (a few hundred microseconds), which gies a high degree of utilization even for short transfers. The analysis also shows that when channels are established very frequently, the signalling capacity limits the performance.

1. Introduction

The next generation of networks will integrate services for different kinds of applications: delay-insensitive asynchronous applications like fax, mail, and file transfer, and delay-sensitive applications with real-time requirements, such as audio and video. These different applications have traditionally been supported by different network technologies. Asynchronous communication

has been provided by computer networks, which are packet-switched and use store-and-forward techniques, like the Internet. Real-time communication, on the other hand, has been provided by circuit-switched, time-division multiplexed telephone networks.

Circuit-switched networks have many attractive properties. Circuits are isolated from each other in the sense that traffic on one circuit is unaffected by activities on the others. This makes it possible to provide guaranteed transfer quality with constant delay, which is suitable for applications with timing requirements. Furthermore, data and control are separated in circuit-switched networks. Processing of control information only takes place at establishment and tear-down of circuits, and the actual data transfer can be done without processing of the data stream, congestion control, etc. This allows large volumes of data to be transferred efficiently [1]. We think that this will be even more important in the future, since developments in photonics will dramatically reduce the cost of transmission, and switches will become the main communication bottlenecks.

The static nature of ordinary circuit-switched networks makes them inappropriate for certain types of traffic. Traditionally, circuits have fixed capacity, long set-up delay and poor support for multicast. These shortcomings make it difficult to efficiently support, for example, computer communication in a circuit-switched network. This has motivated a search for alternative solutions, and the predominant view is that the next generation of tele-communication networks should be cell-switched, based on ATM [2]. Cells are small, fixed-size packets, so this is an orientation towards packet-switching [3]. This means that many of the weaknesses of packet-switching are present also in cell-switched networks, in particular in the area of providing guaranteed quality of service. Therefore additional mechanisms, such as admission control, traffic regulation, scheduling of packets on

links and resynchronization at the receiver are needed to integrate support for different kinds of traffic [4]. One of the main concerns with packet switched networks in general, and ATM in particular, is whether it is possible to realize these mechanisms in a cost-effective way [5], [6], [7], [8], [9], [10], [11].

DTM, Dynamic synchronous Transfer Mode, is a broadband network architecture developed at the Royal Institute of Technology in Sweden. It is an attempt to combine the advantages of circuit-switching and packet-switching, in that it is based on fast circuit-switching augmented with dynamic reallocation of resources, supports multicast channels, and has means for providing short access delay. The DTM architecture spans from medium access, including a synchronization scheme, up to routing and addressing of logical ports at the receiver. DTM is designed to support various types of traffic and can be used directly for application-to-application communication, or as a carrier network for other protocols, such as ATM or IP. A prototype implementation based on 622 Mbps optical fibers has been operational for two years, and work is in progress with a wavelength-division multiplexed version with four wavelengths. An overview of DTM with a more detailed description of the prototype implementation was previously published in [12].

Fast circuit-switching was proposed for use in telephone systems already in the early 1980s [13]. A fast circuit-switched telephone network attempts to allocate a transmission path of a given data rate to a set of network users only when they are actively transmitting information. This means that a circuit is established for every burst of information [14], [15]. When silence is detected, the transmission capacity is quickly reallocated to other users. In the form used in TASI-E [13], fast circuit-switching has been deployed for intercontinental communication between Europe and the United States. Burst switching is another form of fast circuit-switching, where

a burst (consisting of a header, an arbitrary amount of data and a termination character) is sent in a time-division channel of fixed bit-rate and is thus interleaved with other bursts [16]. This makes burst switching different from fast packet switching, where packets are sent one at a time with full link bandwidth. Furthermore, the length of a burst is not, in contrast to a packet, determined before the start of transmission.

It has been shown that the signalling delay associated with creation and tear-down of communication channels determines much of the efficiency of fast circuit-switching [14]. DTM is therefore designed to create channels fast, within a few hundreds of microseconds. DTM differs from burst switching in that control and data are separated, and it uses multicast, multi-rate, high capacity channels to support a variety of different traffic classes. This means for example that it is possible to increase or decrease the allocated resources of an existing channel. Even though a DTM network may have the potential to create a channel for every message, we do not believe this approach suitable for all traffic classes. Rather, it is up to the user to decide whether to establish a channel per information burst or to keep the channel established even during idle periods.

The purpose of this paper is to study the performance of fast circuit-switching in DTM, focusing on the dynamic resource management scheme, with the aim to demonstrate how it can support traffic that consists of short, frequent transfers. The paper is organized as follows. Section 2 gives an introduction to DTM, describes the channel concept and the resource management scheme, and discusses how some of the problems with traditional circuit-switching are dealt with. Section 3 reports and discusses simulation results for various configurations, both for single and multi-hop connections. Finally, conclusions are drawn in Section 4.

2. DTM-Dynamic synchronous Transfer Mode

DTM is designed for a unidirectional medium with multiple access, i.e., a medium with capacity shared by all connected nodes. It can be built on several different topologies, such as ring, folded bus or dual bus. We chose the dual bus, since it has shorter average inter-node distance than a folded bus, and DTM's synchronization scheme was found to be easier to implement on a dual bus than on a ring.

The service provided is based on channels. A channel is a set of time slots with a sender and an arbitrary number of receivers; it is guaranteed that the data will reach the receivers at the rate given by the capacity of the channel. The channels on the physically shared medium are realized by a time division multiplexing (TDM) scheme (Figure 1). The total capacity is divided into cycles of 125 microseconds which are further divided into 64-bit slots.

The slots are separated into data and control slots. Each node has access to at least one control slot, which is used for sending control information to the other nodes. Control messages can be sent upon request from a user, in response to control messages from other nodes or spontaneously for management purposes. The control slots constitute a small fraction of the total capacity, while the major part of the slots are data slots carrying payload. At system start-up, the data slots are allocated to the nodes according to some predefined distribution. This means that each node "owns" a portion of data slots. A node needs ownership of a slot to send data in it, and the ownership of slots may change dynamically among the nodes during the operation of the network.

2.1 Slot Allocation

DTM uses a distributed algorithm for slot reallocation, where the pool of free slots is distributed among the

nodes. There were two main reasons for using a distributed scheme instead of a central pool of slots. First, when a node can establish a channel using only slots from the local pool, there is very little overhead in the channel establishment. Second, a distributed algorithm does not rely on a single node, so it has some tolerance to node failures. The main drawback of a distributed implementation is the overhead of communication and synchronization between nodes.

At the reception of a user request, the node first checks its local pool to see if it has slots enough to satisfy the request and, if so, immediately sends a channel establishment message to the next hop. Otherwise the node first has to request more slots from the other nodes on the bus. Each node maintains a status table that contains information about free slots in other nodes, and when more slots are needed the node consults its status table to decide which node to ask for slots. Every node regularly sends out status messages with information about its local pool of slots. Status messages have lower priority than other control messages so they are only sent when the control slot otherwise would be unused. Furthermore, the reallocation algorithm does not depend on nodes to process all status messages, so a node can safely ignore status messages while it is busy.

The procedure for slot reallocation is simple and works as follows: if a user requests a channel with M slots and the node has N free slots, where , it sends requests asking for slots. The node starts by sending a request to the closest node with free slots. If this node does not have sufficiently many free slots, according to the status table, a request is sent also to the second closest node with free slots, and so on. The node waits until a response for each of the requests has been received and then grants or rejects the channel request, depending on the outcome of the reallocation procedure.

A node that has some amount J of free slots and receives a slot reallocation request for K slots will always give away min(J, K) slots. The node responds by sending a slot reallocation confirmation, which indicates what slots the node gives away. If the requested node does not have any free slots, it responds with a slot reallocation reject instead. In addition to this algorithm, resources may be controlled by a network management system. For example, to prevent the network from starvation, a node can be configured not to give away all its slots, but to keep a certain fraction of its initial share.

2.2 Switching

A DTM network can be expanded by interconnecting several buses with switch nodes (see Figure 2). DTM uses decentralized switching in the sense that any node connected to two or more buses can switch data between them. One advantage with this is that the switching capacity can be increased gradually by adding more switch nodes. The switching is synchronous, which means that the switching delay is constant for a channel. This means that a multi-hop channel has roughly the same properties as a channel on a single bus. The only difference is that a switched channel has slightly longer delay (up to 125 microseconds for each hop). Provided that a switch node can buffer one cycle of data for each of its buses, there cannot be any congestion or overflow in the node.

The synchronization between buses is done on a per cycle basis-cycles are started with the same frequency on all buses. This is accomplished by letting one network node be responsible for generating the cycles periodically on all its outgoing buses. For each new cycle, this node generates a cycle-start marker that is forwarded to all buses in the network. For each bus there is one switch node that is responsible for forwarding the marker onto the bus. Those switch nodes need to be organized in such a way that the marker reaches every bus exactly once. When

the marker reaches a bus, the cycle is restarted on this bus. The reader is referred to [12] for more details on the synchronization scheme.

The cycle time and the slot size are both constant for all buses, which means that the synchronization scheme allows different buses to run at different bit rates. This makes it possible to upgrade or reconfigure individual buses in a network without affecting the rest of the network.

2.3 DTM channels

The channel abstraction in DTM differs from ordinary circuits, in that channels have the following properties.

- Simplex: a channel is set up from sender to receivers. A duplex connection consists of two channels, one in each direction.
- Multirate: channels may consist of an arbitrary number of data slots, which means that channel capacity can be any multiple of 512 Kbps, up to the entire data capacity of the bus.
- Multicast: a channel can have several receivers.

A node creates a channel by allocating a set of data slots for the channel and by sending a channel establishment control message. The control message is addressed either to a single node or to a multicast group, and announces that the channel has been created and what slots it uses.

To create a channel, slots must be allocated at the sender and at each switch node along the channel's route. Thus, switch nodes allocate slots for a channel on behalf of the sender. The switch nodes then start switching the channel, by copying the channel's slots from the incoming to the outgoing bus. An attempt to establish a multi-hop channel fails if any of the switch nodes involved cannot allocate the required amount of slots. In this case another route has to be tried. In a grid structure, there are normally several routes between each pair of nodes. The current

version of the protocol uses source routing [17] together with an addressing scheme based on (x, y) coordinates in the grid. A simple load-balancing scheme for two hops is achieved by letting each switch node use status messages to send information about the amount of free slots on its outgoing buses. For example, there are two possible routes between node 1 and node 4 in Figure 2, so if node 1 wants to set up a connection to node 4 it can choose between using switch node 2 and switch node 3. Node 1 receives status information from node 2 and 3, and can make its routing decision based on this information. This algorithm works well for dense grid networks, where most routes use only two hops, but a general routing algorithm is required for more arbitrary topologies.

2.3.1 Multicast channels

A traditional circuit is a point-to-point connection between a sender and a receiver. DTM uses a shared medium which inherently supports multicast since a slot can be read by several nodes on a bus. A multicast channel can easily be extended to span over several hops, since the switch operation is actually a multicast operation, in the sense that it duplicates data onto another bus (see Figure 3).

3 Network Performance

In this section we investigate throughput and delay under varying traffic conditions. We have performed simulations for two different network topologies:

- A dual bus with 100 nodes.
- A fully connected grid of dual buses with 20 ¥ 20 nodes.

In the simulation model, nodes receive transfer requests from a traffic generator and control messages from other network nodes. These events are put in an input queue at the node, and the node processes one event at a time. The

time to process one event is 5 microseconds. Transfer requests are generated by Poisson processes, and source and destination addresses are uniformly distributed. For each transfer request, a node attempts to allocate slots and, if that succeeds, establishes the channel, transfers the data and takes down the channel. This means that slots are released as soon as the transfer is done.

The simulations are performed for different transfer sizes (1 to 256 kilobytes), for different kinds of user requirements, and for different inter-node distances (0.01 to 10 kilometers). The link bit-rate is 4.8 Gbps, which gives a slot rate of 75 MHz and a cycle size of 9600 slots. Each transfer requests 40 slots per cycle. This corresponds to 20.48 Mbps channels, which means that a 16 kilobyte transfer, for example, takes about 6 milliseconds.

We calculate the throughput by dividing the total amount of transferred user data by the simulated time, and normalize this value to the capacity of one dual bus (9.6 Gbps). The maximum throughput that is possible to achieve is always less than the link capacity, since some capacity is used for control messages. In the single dual bus simulations, with 100 nodes, the control capacity is 5 control slots per node, which corresponds to 5% overhead. The maximum possible throughput is then 0.95.

The grid has more nodes than the single dual bus, but fewer nodes per bus (20 instead of 100). Since the traffic load on a bus is distributed over its nodes, this means that at a given bus load, the grid will have more traffic to and from each node. A node in the grid therefore needs more control slots. However, a node has limited capacity to process control messages, and we have found that with 5 microseconds event processing time, very little performance is gained by using more than 10 control slots per node. We therefore use 10 control slots per node in the grid, which gives a maximum possible throughput of 0.98.

Access delay is the average time from the time that a request arrives to the node until the data transfer starts. It is a measure of the overhead at channel establishment and includes the time it takes to allocate slots, send a channel establishment message to the receiver, and send the first slot of data. In the multi-hop case, the sender waits for a confirmation from the receiver that the channel has been established on both buses before it starts sending data. For the single hop case, the sender alone creates the channel to the receiver, and can therefore start sending data as soon as the slots have been allocated.

3.1 One Dual Bus

The first set of simulations concerns performance of one dual bus. The purpose of these simulations is mainly to study slot allocation schemes for different user requirements. Since slot allocations on different buses are independent operations, these results are generally applicable for the multi-hop cases as well.

3.1.1 Strict Capacity Demand Without Retry

Figure 4 shows the results of a basic simulation where a node makes at most one attempt to allocate the requested capacity for a channel, and rejects the request if the full requested capacity cannot be allocated. Transfer sizes between 1 and 256 kilobytes are simulated. The simulations could not be carried out for the smallest transfers (1 and 2 kilobyte in Figure 4) at high load, due to the simulator event queue growing too large, indicating that the control capacity is exhausted.

At low load, most transfer requests are accepted immediately and the throughput therefore increases linearly with the load. At higher load, slot reallocation becomes frequent, some transfer requests are rejected, and the throughput increases only marginally with the load. If the load gets even higher, the signalling capacity is

exhausted, and throughput does not increase (or goes down, as in the case for 1 kilobyte transfers in Figure 4). Smaller transfers require more frequent control signalling than large transfers at a given load, and therefore throughput is lower for smaller transfers than for larger (at most 0.47 is achieved for 1 kilobyte transfers, which is 50% of the maximum possible throughput, compared to 85% for 256 kilobyte transfers). Throughput is also limited by the strict user behaviour, requiring the entire requested capacity to be allocated in a single attempt. The simulations reported below show that throughput can be increased by relaxing this requirement.

The access delay consists, at low load, mainly of the time it takes for a node to process the transfer request, wait for the first available control slot (for the channel establishment message), and then for the first data slot (80 microseconds together, on the average). When the load increases, nodes have to request slots from other nodes and more delay is introduced.

3.1.2 Strict Capacity Demand with Retry

It is possible to increase throughput by letting a node retry, that is, make more than one attempt to allocate slots for a channel. Figure 5 shows throughput and access delay for different values of the maximum number of retries allowed. When nodes are allowed to retry, more channels can be established and throughput increases (up to 92% of the maximum possible throughput), but at the expense of longer access delay and more signalling. Thus, retry is best suited for applications that have strict bandwidth demand but can tolerate some access delay. However, if a large number of requests are persistently retried (as in an overload situation), performance will degrade. Figure 5 shows that performance decreases when the load is high and nodes are allowed to retry 20 times, which indicates that the signalling capacity is not sufficient for 20 retries.

3.1.3 Flexible Capacity Demand

An application with less stringent demands on capacity may be prepared to accept channels with less capacity than requested. This allows more channels to be established with a single slot reallocation attempt. Figure 6 shows throughput and delay for three cases: when the user is satisfied with any capacity (minimum 1 slot), requires at least half the requested capacity (20 slots), and needs the full capacity (40 slots). Throughput goes up when the user is less demanding. When the user requires only one slot, the throughput achieved is 94% of the maximum possible throughput. The slot reallocation procedure is however the same in all three cases, which explains why access delay is practically the same in the three cases.

3.1.4 Performance as a Function of Distance

When the distance between nodes is increased, it takes longer time to exchange control messages for slot reallocation. This can be seen in Figure 7, which shows throughput and access delay for different bus lengths (with strict capacity demand without retry, i.e., the same slot allocation principle as in Figure 4). The access delay increases significantly when the bus gets longer. However, this affects mainly the creation of channels, and throughput is therefore relatively independent of distance. Another effect of increased bus length is that it takes longer time to propagate status information, which may increase the probability of slot reallocation failure. This would mainly influence throughput, but the simulation results indicate that this has only a minor effect.

3.2 Grid Network

Figure 8 shows the result of a simulation for multi-hop channels. The network is a fully connected grid network with 20 ¥ 20 nodes. Channels use at most two hops, and the routing decision is based on the information from status

messages. The slot allocation principle is the same as in Figure 4, that is, strict demand without retry.

With uniform distribution of source-destination pairs, the theoretical maximum throughput of a fully connected grid is, where n is the number of buses. With 2.1% signalling overhead, the 20 ¥ 20 grid has a maximum possible throughput of approximately 20.6. Figure 8 shows that for 256 kilobyte transfers, the maximum throughput is 97.5% of the possible maximum, compared to 95.1% for 16 kilobyte transfers. This is significantly more than in the case of one dual bus (Figure 4), and our explanation for this is that there are less nodes per bus in the grid, which means that the pool of free slots is less scattered, and therefore there is a higher probability that slot reallocation succeeds.

The access delay for a multi-hop channel will be longer than on a single hop, since slots have to be allocated on two buses. For 256 kilobyte transfers, the access delay is roughly 50 per cent longer in the multi-hop case compared to the single hop case. One could expect the access delay in the grid to be longer than this. There are two main reasons why this is not the case. First, there is a certain amount of parallelism in channel establishment over two hops. Second, the interval between control slots is shorter in the grid, so a node spends less time waiting for control slots. For 16 kilobyte transfers, however, the delay increases dramatically for higher load, which indicates that signalling capacity is insufficient.

4 Conclusion and Future Work

The Dynamic synchronous Transfer Mode (DTM) is a network architecture for integrated services. It is based on fast circuit switching, and provides multi-rate, multicast channels with short set-up time. A channel gives guaranteed capacity and constant delay, which makes it well suited for real-time traffic, such as video and audio. In contrast to traditional circuit-switched

networks, DTM also provides dynamic reallocation of resources between nodes in order to support dynamically varying traffic.

We have reported performance analysis through computer simulations for various configurations. The analysis focuses on packet-like traffic patterns, since this type of traffic is considered to be the most challenging for a network based on circuit switching. The intention is to study how network utilization and access delay are affected when channels are frequently established and released. We use 20 Mbps channels, where a channel is established for each transfer, and the transfer size varies between 1 and 256 kilobytes. The analysis was performed for two kinds of topologies: a dual bus and a grid of dual buses. The protocol requires that a certain fraction of the total capacity is reserved for control information: we used about five per cent for the dual bus, and two per cent for the grid.

The results show that when the traffic consists of short, frequent transfers (of a few kilobytes), performance is determined by the amount of signalling capacity available. For the dual bus, good results are achieved even when the network is loaded with transfers that are only few kilobytes long, whereas the grid was saturated for 16 kilobyte transfers. To summarize, the paper shows that fast circuit switching, as applied in DTM, performs well even for packet traffic using short-lived connections. This, in combination with its inherent support for real-time traffic, makes fast circuit-switching an appealing alternative for B-ISDN and beyond.

For future work, the performance results suggest that in order to send packets on DTM, one needs to find a scheme for multiplexing packets onto channels that groups packets together into slightly larger units. Assuming that computers generate traffic that conforms to the train model [18]-bursts of packets where many of the packets within a burst have the same destination-a simple scheme

similar to those used in the early fast circuit-switched networks seems appropriate [13], [15]. In such a scheme, the channel is closed and its resources are released when the sender has been idle for more than a certain period. This is, however, an area that needs further investigation. The performance results are encouraging and we will explore further the effects of non-uniform traffic models, such as bursty sources and asymmetric sender-receiver distributions. In addition, we will implement and investigate mechanisms such as fast channel creation [19], [20], [21] and slot reuse [22] in the simulator.

B. Performance Analysis of Slot Management in DTM Networks

Abstract

This paper presents performance results and analysis of a Dynamic synchronous Transfer Mode (DTM) network, a new high-speed network architecture based on fast circuit switching. The performance analysis considers utilization, delay and blocking. The analysis is biased towards packet switching, which is considered the most difficult service to provide in a network based on circuit switching.

Resource management uses tokens to guarantee conflict free access to time-slots. The DTM model includes two different token management schemes. The first is an asymmetrical scheme based on a central token manager that manages all free tokens. The second is a symmetric scheme based on a distributed token manager where all nodes are sharing the token pool. Tokens in the distributed scheme are passed by a reallocation protocol. A token fragmentation problem is identified for the distributed scheme and a solution based on a defragmentation mechanism is presented. Both models support slot reuse and a scheme for fast connection establishment.

Traffic generators for the simulation are generating traffic with exponential inter-arrival distributions, bursty arrivals or asymmetrical client-server like traffic. Results show that the connection establishment overhead is low enough, that utilization may be high and that the reallocation protocol works well, even if a circuit is established for each individual packet. When the slot reuse scheme is turned on, utilization increases by almost a factor of two and other performance metrics improve as well. Finally, the results identify signaling capacity as the most significant limiting factor to achieve high performance if the average packet size is small.

1. Introduction

New high-capacity communication networks and protocols are constantly being developed by the communications industry and academia. This development changes frequently and new results are important to application developers who integrate real-time audio, video, and asynchronous communication services into applications. The applications can be found on a wide range of network access terminals. Terminals act as network hosts and may be almost any electronic device, including small pocket telephones or television sets, multi-media workstations, and million-dollar supercomputers. Hosts differ by orders of magnitude in their needs for processing power, and in their communication services requirements. These disparate needs are currently reflected in a set of independent network classes. Each network class is optimized for its particular traffic and applications: cable television networks use unidirectional broadcast networks where the capacity is divided into fixed-sized subchannels carrying video. A telephone network uses only 64 kbit/s duplex circuits with guaranteed throughput and tightly controlled delay variations. Computer networks, such as the Internet, allow a large number of parallel network

sessions by use of connectionless packet switching. They also use statistical multiplexing to efficiently utilize links. A backbone network for mobile systems needs extra control (or signaling) capacity to dynamically track all active terminals.

With this wide spectrum of applications existing today and to be augmented in the future, it is infeasible to continually invent new, sometimes global, networks and a new terminal interface for each new type of service. Instead, an integrated services network supporting existing and new services needs to be deployed. The overall goals for such a network are scalability to global size and maximum sharing of expensive network components to minimize cost. Optical transmission technology has been shown to provide the necessary link capacity at a low enough price to make integrated services networks a realistic solution.

A new integrated optical network with much higher capacity will, however, introduce new problems not seen in today's more specialized and lower performance networks. First, when network capacity increases and the information flight latency remains limited by the speed of light, the increasing bandwidth delay product will put higher demands on mechanisms that isolate a user's traffic from third-party network traffic. A telephone session, for example, should not be affected by another user opening a high-capacity video channel. Second, applications and protocols will need to operate reliably with an increasing amount of information in transit to benefit from the increased network capacity. This will lead to larger burst and transaction sizes in the network.

Current networks using stateless packet switching protocols such as the Internet Protocol (IP)[rfc791, Come91:Internetworking1] have turned out to be extremely scalable. They have evolved from small networks connecting just a few Defense Advanced Research Projects Agency (DARPA) [rfc1120,Come91:Internetworking1] research

computers in the mid seventies to the current global and ubiquitous Internet[rfc1118]. Shared medium local area networks (see [stallings94:data]) (LAN's) such as CSMA/CD, token ring and FDDI are used in the Internet as simple building blocks connected by routers or bridges. The combination of easy expansion, low incremental node cost and tolerance to faulty nodes has resulted in simple, flexible, and robust networks. Also, the shared medium allows efficient application of new multicast protocols such as IP multicast[rfc988].

A drawback of the shared medium is that it typically permits only a single terminal to transmit at any time, thereby not utilizing all network segments efficiently. A scheme allowing the capacity of the medium to be reused may be designed [7,8,9,10,11], but this is often at the cost of complexity in the high-speed access control hardware. Access control mechanisms for a shared medium also depend strongly on the size of the network and are usually efficient only for local area environments.

The two main types of network are connection oriented circuit-switched networks used for telephony, and connectionless packet-switched networks exemplified by the Internet. When a circuit-switched network is used for data communication, circuits need to remain open between bursts of information, wasting network capacity. This problem arises because circuit management operations are slow compared to dynamic variations in user demand. Another source of overhead in circuit-switched networks is the limitation of requiring symmetrical duplex channels, which introduce 100% overhead when information flow is unidirectional. This constraint also makes multicast circuits inefficient and difficult to implement. A connectionless packet-switched network, on the other hand, lacks resource reservation and must add header information to each message before transmission. Furthermore, latency in a connectionless packet-switched network cannot be accurately predicted and packets may even be lost due to

buffer overflow or corrupted headers. The latter two factors make real-time service difficult to support. Congestion avoidance mechanisms[12] can isolate traffic streams of different users. These schemes are, however, limited to operating on a time scale comparable to the round-trip packet delay.

To address the problems mentioned above, the communications industry is focusing on the development of Asynchronous Transfer Mode (ATM) [13]. ATM has been proposed for LAN's and many future public networks. International Telegraph and Telephone Consultative Committee (CCITT) also adopted it to be used as the transfer standard in broadband ISDN (B-ISDN) [14]. ATM networks are connection-oriented and establish a channel just as circuit-switched networks, but use small fixed-sized packets called cells for information transfer. The packet-switched nature of ATM means that the network needs to have new mechanisms such as buffer resource managers and link schedulers to establish real-time guarantees for a connection.

Our solution for providing real-time guarantees centers on a circuit-switched network and we must therefore address circuit-switching concerns such as those described above. We also employ a new shared-medium control protocol, and so must consider common shared-medium problems. Our design, called Dynamic synchronous Transfer Mode (DTM) [15,16,17], uses channels as the communication abstraction. Our channels differ from telephony circuits in various ways. First, establishment delay is short so that resources can be allocated/deallocated dynamically as fast as user requirements change. Second, they are simplex and so minimize overhead when the communication is unidirectional. Third, they offer multiple bit-rates to support large variations in user capacity requirements. Finally, they are multicast, allowing any number of destinations.

DTM channels share many beneficial properties with circuits. There is no transfer of control information after channel establishment, resulting in very high utilization of network resources for large data transfers. Support of real-time traffic is natural; there is no need for policing, congestion control or flow-control within the network. The control information is separated from data, which makes multicast less complex. The transmission delay is negligible (i.e. less than 125 µs) and there is no potential for data loss caused by buffer overflow as in ATM. Bit-error rates depend on the underlying link technologies, and switches are simple and fast due to strict reservation of resources at channel setup. The aim of this paper is to study the performance of DTM in the areas where traditional circuit-switched networks fall short: dynamic bandwidth allocation, channel establishment delay, and as shared media networks. Principles for resource management (called token management) are presented and evaluated. We report results from simulations where we expose DTM to traffic patterns that are more like the relatively short-lived transfers (4-4000 kbyte) seen in computer communication. Traffics are with bursty inter-arrivals, client-server oriented as well as with exponentially distributed inter-arrival times.

Section 2 describes the DTM protocol and its channel concept. In section 3 the token protocol is described. Section 4 follows with a discussion of the simulation set-up, and Sections 5 presents the simulation results for various configurations. Finally, in Section 6 conclusions are drawn and future work outlined.

2. The DTM medium access control protocol - DTM-MAC

The basic topology of a DTM network (see Fig.9) is a bus with two unidirectional optical fibers connecting all nodes. Several busses with different speeds may be connected together to form an arbitrary multistage

network. In the current prototype implementation [15], buses can be combined into a two-dimensional mesh [16]. A node at the junction of two buses can synchronously switch data slots between the two busses. This allows for efficient switching with constant delay through the node. The primary communication abstraction in DTM is a unidirectional, multi-rate and multicast channel [18].

The DTM medium access protocol is a time-division multiplexing scheme. The bandwidth of the bus is divided into 125 us cycles (Fig.[10]), which in turn are divided into 64-bit time slots (or slots for short). The number of slots in a cycle thus depends on the network's bit-rate; for instance, on a 6.4 Gbit/s network there are approximately 12500 slots per cycle.

The slots are divided into two groups, control slots and data slots. Control slots are used to carry messages for the network's internal operation, such as messages for channel establishment and bandwidth reallocation. The data slots are used to transfer user data and are not interpreted by intermediate network nodes. Intermediate nodes are nodes between the source and destination.

In each network node there is a node controller, which controls the access to data slots and performs network management operations, such as network start-up and error recovery. The main tasks of the node controller are to create and terminate channels on demand from users, and to manage network resources in response to user requests and in the background.

Control slots are used exclusively for messages between node controllers. Each node controller has write permission to at least one control slot in each cycle, which it uses to broadcast control messages to down-streams nodes. Since write access to control slots is exclusive, the node controller always has access to its control slots, regardless of other nodes and network load.

The number of control slots a node uses may vary during network operation.

3. Token management

The majority of the slots in a cycle are data slots. Access to data slots changes over time, according to traffic demands. Write access to slots is controlled by slot tokens (or tokens for short). A node controller may write data into a slot only if it owns the corresponding token. The token protocol guarantees the slot access to be conflict free, which means that at most one node writes data into the same slot.

Control messages for channel establishment and bandwidth reallocation carry sets of tokens as parameters. However, a control message is 64 bits and can therefore have only a small number of parameters. This means that if a user requests a large bandwidth transfer, it may be necessary to send several control messages to create the channel. This introduces extra access delay and consumes signaling capacity.

We are considering several mechanisms to decrease the amount of information that needs to be sent during channel creation and token reallocation. The first optimization in token management is to introduce block tokens. A block token is transferred in a single control message and represents a group of tokens, but can only be used for particular combinations of tokens. For instance, in the network simulations described in this paper, a block token is denoted by a slot number and an offset giving the number of contiguous slots in the group. The block token optimization assumes that the token pool is not fragmented into small pieces. The amount of small token blocks in the free pool, may be a problem and will be referred to as fragmentation.

3.1 Slot reuse

The token protocol guarantees that a data slot can never be used by two nodes simultaneously on the bus. Sometimes this protocol is too conservative. Fig[11] shows an example of how three tokens (A,B, and C) are reserved for three channels. Nodes are connected by bus segments and channels typically use a subset of the segments on the bus (gray color) and the rest are reserved (white color) but left unused and thus wasting shared resources. A better alternative is to let channels only reserve capacity on the segments between the sender and the receiver as the example in Fig[12]. A single slot may in this case be used multiple times on the bus. Channel D and E are using the same slots as channel A and C but on different segments. This is referred to as slot reuse. Slot reuse enables simultaneous transmissions in the same slot over disjoint segments of the bus.

Slot reuse is a general method to better utilize shared links in ring and bus networks. The slot reuse algorithms in DQDB [11,7], Simple [19] and CRMA [20] depend on control information in the slots. Buffer insertion networks [10,8] when combined with destination release as in METARING [9] may reuse capacity of individual links and resolve eventual contention by delaying the incoming packet stream through an elastic buffer.

With slot reuse, the access scheme complexity is increased, regardless of whether it is done in hardware as in DQDB, Simple, and CRMA, or in software, as in DTM. When implemented in systems other than DTM, slot reuse also adds complex hardware to the critical high-speed path through a node and therefore increases the node delay.

To allow slot reuse in DTM the block token format is extended to include parameters describing the segment(s) it is representing. The token management protocol is also modified to avoid conflicts in the slot number dimension as well as the segment dimension. The most important assumption is that no hardware changes to the original prototype implementation were allowed or needed. The

performance gain is also quite clear: on a dual-bus where source and destination pairs are uniformly distributed, it has been shown [21] that throughput may be increased by a factor of two. The potential drawback of slot reuse in a DTM network is the higher algorithm complexity and potentually higher load on the node controller and signaling channels (especially if the average channel duration is short).

3.2 A central token manager

Two token management schemes were evaluated. The first and simplest is to let a single node controller manage all free tokens on a fiber. This sort of centralized server mechanism has also been used in systems such as CRMA [22], where the head-end node distributes the fiber capacity to all other nodes. We configured the simulator so that for each fiber, a node one third from the slot generator is the token server. A token server will then have about the same amount of traffic on both sides.

Each time a user request arrives at a node, the node first requests tokens from the manager and then locks them throughout the channel life-time. When the user issues a request to disconnect the channel, the tokens are immediately returned to the manager. All requests are delayed during the token request and accesses are serialized through the central manager.

3.3 A distributed token manager

The distributed token manager is fundamentally more complicated than the centralized. We tried to keep it as simple as possible. In our method each node regularly broadcasts status information about how many free tokens it has. The other nodes store this information in their status tables. A node that wants more capacity consults its status table to decide from which node to request slots. The protocol on the initiating side works as follows, when a user request arrives to a node:

1. If the node has sufficiently many free tokens to satisfy the request, it allocates the requested amount of slots to the user, and starts the channel by sending a channel establishment message to the destination node, and then transmitting data using the reserved slots.

2. Otherwise the node marks its available tokens as being reserved, and then checks its status table: if the total amount of free tokens in the network is not enough to fulfill the request, the request is rejected (blocked). Otherwise the node requests tokens from nodes with unused capacity.

If one of these nodes receiving a token request does not have the requested amount of free slots, it still gives away all slots it has. In any case, it sends a response back to the the requesting node. A node fulfills incoming requests in strict FIFO order.

When a node receives a response to a token request, it marks the slots it receives in the response (if any) as being reserved. When the node has received responses to all requests it has sent, it either starts the channels or rejects the user request, depending on if it has acquired sufficient capacity. If the user request is rejected, the reserved slots are marked as free.

At startup all free tokens are evenly distributed among the network nodes and each node takes at least one of its free tokens, moves it (them) to the active state and declares it (them) to be a control slot. User requests may now be accepted and tokens can be moved between nodes on demand. When the local node contains enough tokens to fulfill an arriving user request, the request may be accepted without any token reallocation.

A weakness with this scheme is that slot reallocation is only triggered by user requests, and user requests are delayed by pending token reallocation. An optimization we have implemented to remedy this is to also perform slot

reallocation in the background. This results in less need to reallocate tokens for small to medium sized requests.

The pool of the free tokens can be distributed in other ways than uniformly to increase the speed for succesful reallocation and usage. If fewer nodes are managing the pool, the channel blocking is decreasing as a result of lesser possibility for tokens to be wringly reallocated.

The complete token pool is in this case proportionally distributed (nodes near the slot generator will have more tokens than nodes far away from this) among all nodes. The transfers of nodes can occur between each pair of nodes in place of always use the server node. When the local node comprises sufficient number of tokens to satisfy an arriving user request, the request can be accepted without any token reallocation. Further, as long as the arriving user requests are well matched by the pool distribution, no reallocation will ever be necessary.

Several questions need to be answered before deciding how to distribute the token pool. We address the following issues:

1. When local resources in the node are not enough to satisfy a user request, which other node should be asked for more tokens?

2. If a node asks for tokens from several nodes, how many tokens is it going to ask for and should the node reject the channel if it receives a fraction of the requested capacity?

3. If tokens move freely among the nodes, will the token pool be fragmented into small pieces and render the block token optimization scheme useless?

3.3.1 Status messages

We decided to use status messages to distribute information about the pool of free tokens. Status message information is used to help a node choose a suitable node

when asking for more resources. This method addresses the first question above.

Our scheme works as follows. Each node regularly broadcasts status information about how many free tokens it has. The other nodes store this information in their status tables. A node that wants more capacity consults its status table to decide from which node to request slots. The broadcasted status information gives an approximate view of the current state of token information, so token requests may be rejected because they were sent to nodes that no longer have tokens to give away.

Status tables are "soft" information in the sense that the system will work even if they are out of date or unavailable. They should, however, improve the success rate of reallocation procedures.

3.3.2 Avoiding unnecessary rejects

When comparing base performance of the centralized (fig. 17) and the distributed (fig. 20) token management we can see that there exists a new type of reject of user requests which are frequent in the distributed version when there still is unutilized resources in the system.

A node uses the status table to pick node(s) from which to request tokens. When the request reaches the target node the amount of available capacity may have changed and a smaller amount than requested may be returned to the requesting node, resulting in a user rejection.

This behavior results in unnecessary token transfers and waste bandwidth resources as slots are unused during the token transfer. The status message mechanism will also work less efficient when tokens are moved more frequently.

If the pool is distributed proportionally among a great number (hundreds) of nodes, the size of the average pool will be relatively small. When the load is high, the number of free tokens in the pools will decrease even further. If nodes also create and destroy channels at a very high speed, the amount of free capacity in individual nodes will vary between a little amount of capacity and no capacity at all. If now the average capacity which is requested by a user is large in comparison with the number of free tokens of a node, then more nodes must be asked to fulfil the request. At this time the possibility that the requested nodes have no free capacity will lead to a user reject.

There are many ways to address this problem without going back to a centralized model. Firstly, we do not need to give away any slots at all if the complete request cannot be fulfilled. This protocol is applicable even if only one node is asked about free tokens, but if more nodes are asked it can still result in the movement of tokens or that tokens are locked unused. Secondly if we, after a token request, receive fewer tokens than the number requested, we can simply once again and many times try to request the procedure for token request. This will increase the possibility for a user request to be accepted and that the received tokens will be used. The expense for the repeated examination will be increased signalling and increased access delay and can deteriorate the performance on an overloaded network. Even the repeated attempts by the user leads to longer set up delays for a request which has been submitted repeated examination. Thirdly, the user can sometimes wish to accept a channel with lesser than the requested capacity instead of it being rejected. If e.g. a user receives 50% of what has been requested, he

can be willing to accept. In fig. 13 there is disclosed the performance for small (16 kbytes) user requests with different least acceptable capacities [100% (40 slots), 50% (20 slots), and 5% (1 slot)]. A lower average, least acceptable bandwidth will result in a higher utilization. In fig. 14 there is disclosed performance which will be the result if the user is doing repeated attempts up to 8 times before we finally block the request. The utilization will increase (and the blocking will increase) at the expense of more signalling and longer delays. Repeated examination a great number of times will counteract the productivity if it occurs often.

Absolutely, the yield with a flexible user request policy is in lower possibility of rejection and higher total throughput. Either of the configurations disclosed in fig. 13 and fig. 14 can be decided upon at the time when a request is arriving. A user having strict demands on the capacity on a channel can request repeated examination until sufficient capacity is allocated, but another user rather can accept a channel with lesser than the requested amount. For the rest of the simulations presented herein, we define the lower acceptable bandwidth to be 50% of the requested capacity.

3.3.3 Fragmentation

In the general case, the average number of contiguous free blocks in a node is small, due to the random movement of tokens and the varying capacity of users request. This fragmentation renders the block token optimization practically useless, and the access delay is relatively long (milliseconds) for high capacity channels. To make block allocation efficient it is necessary to reduce fragmentation of free tokens, otherwise fragmentation will be by far the main contributor to access delay for high bandwidth channels at moderate to high load. Low capacity channels will almost always have a very short channel establishment delay independent of the current amount of

fragmentation. In the case of slot reuse, the fragmentation problem is even worse, as fragmentation may occur in both slot (time) and segment (space) dimensions (see Fig.12). This is in the centralized server version a particular application of the general dynamic storage-allocation problem [23]. In the distributed token manager most of the fragmentation is a result of using many free pools (one for each node). Two free adjacent tokens can only merge if they are found in the same node.

We have implemented a mechanism called the defragmentation scheme that tries to avoid fragmentation if possible and increases the average block size of free tokens in the nodes. The scheme is used both with and without slot reuse.

Our scheme works as follows:

1. Define a home node for each token at network startup and distribute tokens in such a way so tokens sharing the same home node will always define a continuous slot range. This results in a large average token area in the token map shown in Fig.12.

2. When two slot-consecutive tokens with the same slot or segment range exist in the free pool, merge them into a single token (sometimes a recursive split and merge operation is needed). When doing a merge, always prioritize segment merge before a slot number merge.

   The reason for this is that tokens spanning over just a few segments are less useful to other nodes than tokens spanning over many segments.

3. When a node gets a token request from the local user or a remote user, pick a token from the token pool using a best-fit algorithm in slot number and segment number dimension (see Fig.12). The value of a token is calculated as the area of a token in the token map and we try to pick the token with the smallest area that fulfill the requested capacity.

4. When a node needs to request tokens from other nodes, it does not ask for small chunks from several nodes if it is possible to ask for larger chunks from fewer nodes. The status tables provide this information. Transfer of tokens is therefore more efficient, and there are fewer establishment messages and less fragmentation.

5. Free tokens are sent back to home nodes when they have been idle for a significant time or after a long transfer.

This scheme returns tokens to "home" nodes as a way to increase the probability that two consecutive tokens can be merged in the free list. If home node ``gravity'' is too strong, the scheme will result in less sharing of resources and unnecessary signaling. If it is too weak, fragmentation will still remain a problem.

To evaluate the defragmentation mechanism we performed another set of simulations. We designed three different simulators [A, B, C]. The simulator A was configured to not have any fragmentation at the start time for simulation and to use the defragmentation plan described above, B started with maximal fragmentation of the complete resource pool. All the tokens had only one slot and no tokens in the home nodes were connected before the defragmentation mechanism. Finally, the simulator C was started without the use of the defragmentation mechanism and with the pool which has the maximal fragmentation. In all the cases the slot reuse was switched on and the load was defined to be 80%.

Access delay as a function of simulated time is disclosed in fig. 15 for a network of 10 km. The simulator C started with a long access delay and the delay increased when the signal channels were overloaded and the message queues increased. The simulator B utilizing the defragmentation mechanism starts exactly as badly as C, but already after 10 ms the average access delay is below 500 ms. Later,

when a second of simulated time had elapsed, the B-curve is connected in the most nearest with A, i.e. it converges if the performance of the simulator is started without any fragmentation at all. The converging speed depends on the amount of free capacity in the network and consequently on the load. The load during all these simulations was 80%. The defragmentation mechanism clearly improves the access delay and does even do the block token optimization meaningful in the distributed realization.

3.4 DTM performance

We are mainly interested in two kinds of performance measures in this paper: utilization and access delay. Utilization is the portion of the nominal network capacity that is actually used for data transfer, and is a measure of the efficiency of the network. Access delay is the time from the arrival of a user request to the sending of the first data of the request, which we think is an important measure for how well computer communication traffic can be supported.

There are two main factors that influence utilization in DTM. First,each node is assigned signaling capacity in the form of control slots, which means that there are fewer slots available for data transfer on a bus with many nodes, given a fixed link capacity. Secondly, token reallocation incurs overhead since while a slot token is being reallocated between nodes, the corresponding slot cannot be used for data transfer.

Access delay depends mainly on the load on the control slots, and on how many control messages that need to be sent to establish a channel. The access delay is typically a summation of several delays (and typical values): Node controller processing delay [5 us], delay in finding and allocating free tokens [100 us], waiting for the first available control slot to pass [50 us], and finally waiting for the first allocated data slot to be filled with user data [62.5us]. In addition, messages are

delayed in queues at the input to node controllers waiting to be processed. In the simulations presented in section 5.2 the average delay is up to a few hundred microseconds.

4. Network simulations

In the simulation model, each transfer starts with the arrival of a new "packet" of information. The node controller tries to allocate resources for the transfer, transmits the packet and finally releases the channel. This is a simplification of the mechanisms in a real system, where channel set-up, data transfer and channel teardown are independent operations initiated by the user. For example, a user that knows that a transfer is about to take place may "hide" the channel establishment delay by requesting a channel in advance, so that it is already established when the transfer starts. Between a setup and teardown, the capacity of the channel is completly reserved for the user. The most straight-forward use of the channel is for a single transfer, such as a file transfer or a video transmisson.

Depending on the characteristics of the application, it is possible to optimize the usage of the channel. For instance, a channel may be used to transfer sequences of higher level messages such as ATM cells or IP packets.If it is a multicast channel, messages to different destinations can be multiplexed onto it. This means that each message will reach every receiver on the multicast channel and receivers must be able to filter messages. An alternative solution is to create and destroy a channel for each message, but reserve the tokens between messages so that the tokens are readily available for the next message in the sequence. We do not incorporate this type of user behaviour in the simulations, since they are optimizations for particular applications. Instead we focus on how the network performs without user level optimizations.

The sender may start sending data as soon as the resources are allocated, even before the receiver receives the channel establishment message. This is called fast channel establishment [24]. The receiver will eventually respond with a control message to accept or reject the channel.

User requests have the following parameters:

* Packet size which is the amount of user data transferred between channel establishment and channel release. We simulate packet sizes from a few kbytes up to a few Mbytes.
* Requested capacity for a channel, which is the number of slots that a node tries to allocate. For all the simulations in this paper the requested capacity is fixed to 40 slots or 20.48 Mbit/s.
* Minimum acceptable capacity. A node blocks a request if it cannot allocate this number of slots. This is normally set to 40 or 20 slots (100% or 50% of requested capacity).
* Source address.
* Destination address.

The source and destination addresses are generated randomly (all nodes with the same probability) and user inter-arrival times are exponentially distributed. The simulations investigate the effect of signaling capacity and slot reallocation overhead on utilization, channel set-up delay, and blocking. We simulate a topology with the following characteristics:

* A dual bus network with 100 nodes. Even though it is theoretically possible to connect many more nodes to a bus, we think that management of networks with more than 100 nodes on a bus may be infeasible. With 100 nodes the capacity sharing is significant enough to exercise and test the token management protocol.
* The capacity of each bus is 6.4 Gbit/s. We believe such a capacity is realistic for what is realisable within a

year or two; 2.4 Gbit/s optical links have been available for a few years, and 10 Gbit/s links have been announced to appear soon on the market. 6.4 Gbit/s corresponds to 100 MHz slot rate, which is the rate at which the slot processing MAC hardware would operate. 100 MHz is attainable with current CMOS technology.

* The total signaling capacity is the same for all nodes, but the slots are partitioned between the two fiber directions proportionally, depending on where the nodes are located on the bus. The closer a node is to the slot generator the more control capacity is needed. The sum of the control capacity on both buses will however be the same for all nodes. In the network with two token servers, the servers have more control capacity and higher processing capacity than the other nodes.

* The length of the bus is 10 km, which gives a large enough network that the effects of propagation delay are not negligible. We further study the effects of propagation delay in the simulations in Fig.19 and Fig.21, which use bus lengths of 1 km, 10 km, 100 km and 1000 km.

* Two different token management schemes are simulated: an assymetrical scheme where all the tokens on one fiber are managed by a single token server and a symmetrical scheme where each node controller manages a small piece of the global token pool.

5. Performance

5.1 Ideal protocol

When analyzing the performance of the DTM dual-bus network, the issue of maximum theoretical performance must be addressed and compared to the simulated performance. The maximum theoretical performance is also used in this paper to compare the different schemes and implementations we are evaluating.

The maximum throughput of a dual-bus system without slot-reuse, can be defined as twice the link capacity, given that both fibers receive the same traffic. In a system with slot reuse the system throughput also depends on the source destination distribution. To obtain this throughput for the dual-bus we used a Monte Carlo simulation where source and destination addresses were uniformly distributed (see left graph Fig.16). In the right graph in Fig.16, performance of a DTM network is included. The DTM network uses a centralized token manager and users request to transfer 4 Mbyte of information each time. In this system, signaling capacity is not a bottle-neck and utilization is found to be close to the ideal case. Real traffic behaving like this is bulk-data transfers and audio/video streams. The small differences that are shown are results of: First, some capacity is in DTM used for control slots lowering the available number of slots to be used by data transfers. Second, random generators for the DTM simulations do not generate exactly the same amount of traffic in upstream and downstream directions, and may result in blocking in one of the directions when capacity is available in the other. Third, during channel establishement resources may be locked unused momentarily, wasting some capacity.

5.2 Central token manager

In the case of a central token manager the two managing nodes need more signaling capacity than other nodes (we assign 8 times as many control slots to a server node than to other nodes).

Results of the first set of simulations are presented in Fig.17. Users request 20 Mbit/s channels, inter-arrival times are exponentially distributed (generated by a Poisson process) and the simulations are performed with different packet sizes. If the full capacity of the channel cannot be allocated, the request is rejected and

tokens are returned to the token server. Packet sizes vary from 4 Mbyte to 2 kbyte, at which point we begin to see throughput degradation.

Throughput degradation may occur if processing capacity in nodes or control channel capacity is too small. The server nodes may be especially overloaded. The result is that queues containing control messages start growing very big. The control tokens represent unused capacity, and therefore the throughput is degraded.

In the 4 kbyte per channel simulations, the control capacity is the limiting factor and if more control slots (signal capacity) are added, 4 kbyte and even smaller packets can be more efficiently supported.

The next set of curves, in Fig.18, show how the slot reuse mechanism improves the performance of the system. Throughput increases by almost a factor of two before any significant number of channels are rejected. The uniform source and destination distributions of channels limit the amount of increased capacity gained by slot reuse. It has been shown that if the source and destinations are generated evenly, as we do, throughput may be doubled on a dual bus [21]. In the simulations it may also be seen that beyond an offered load of 2.5 we may actually get throughput higher than 2.0. This level of throughput, however, cannot be reached without some channels being rejected. The channels with the highest probability of being rejected are the ones that use many slots or segments. Therefore the system "filters" less greedy user requests and throws away the rest. This is normally not an acceptable behavior and we therefore do not investigate this any further.

Throughput degradation occurs at an offered load of 1 with the 4 kbyte transfers. Even if we have enough resources in the token server, we cannot establish and destroy, channels fast enough as the control channel is congested. Further, we also see a throughput degradation of the 8

kbyte simulations at an offered load of 1.8, for the same reason.

It may be concluded from the simulations in Fig.18 that the slot reuse mechanism almost doubles the system throughput with only minor changes to the centralized token protocol, as long as the control and server processing capacity is not a bottleneck. From the curves it can also be seen that access delay actually decreases when the load increases from 0.1 to 0.5. This is a result of how slots are assigned to a channel and is not related to the token request procedure. The time it takes to request tokens from the server is strictly increasing with the load.

When comparing the DTM performance in Fig.18 and the theoretical values in Fig.16, we see that even short bursts (a few milliseconds duration) can be supported efficiently.

5.2.1 Performance of the central token server as a function of bus length

When a single token server is used, each channel establishment requires a token to be requested from the server. If the length of the bus is increased, the token request will take a longer time and may therefore also limit the throughput and increase the access delay.

In Fig.19 we increased the length of the bus by a factor of 100 to 1000 km (node to node delay is now 50 μs). Both access delay and throughput may now be limited by the round-trip latency to the token server.

Access delay in this case depends on the distance to servers, but is independent of the tran sfer size. Throughput depends strongly on the average transfer size as the establishment phase is amortized over the data transfer phase.

Channels transferring large amounts of information such as 256 kbyte with a duration of one tenth of a second are still efficiently supported when the bus length is 1000 km.

5.2.2 Discussion

Utilization of a centralized token manager has several benefits. Clients may be simple as they only contain state information related to their own opened channels. Slot reuse is also simple and efficient, as the slot server has all of the free tokens to choose from when trying to satisfy a user request. The server may also implement other policy related functions such as admission control and fairness. The fragmentation of the free token pool in the server is normally very modest, resulting in very few connection establishment messages per channel even for high capacity user requests.

There are also drawbacks. A user that frequently establishes and destroys channels may introduce excessive

signaling by always returning tokens after use, but then requesting the tokens again within a short time period. The processing capacity of the server node may get overloaded if there are many nodes on the bus or if the average packet size is very small. If the media length is very large relative to the product of bit-period, bits-per-packet and media velocity, the round trip time to the server may also limit performance. Finally, the server node contains state information that all nodes depend on to create channels. A server node failure may therefore affect all the nodes.

5.3 Distributed token control performance

In this section we simulate and investigate the properties of a fully distributed token manager.

5.3.1 Distributed token manager

When evaluating the performance of the distributed token manager with slot reuse and defragmentation, we used the same traffic and parameters as in Sect. 5.2, but used a policy whereby requests are accepted if 50% of the requested capacity can be allocated.

The results presented in Fig.20 are from the simulations with slot reuse, a fully distributed token manager, status messages which describe how much capacity a node possesses, and the defragmentation scheme. All the nodes have the same process capacity and the process load is much lower than the load the servers in fig.18 are receiving. The dependences between nodes are also much lower, which results in a higher extent of reliability. The system performs better than any system without slot reuse, but not as well as the centralized system described earlier.

Compared to the centralized scheme (see Fig.18) blocking is higher and starts at a lower load.

A not expected result was that the performance in reality decreased when the packet size increased! After once again having checked up the results we found that a large average transferring size results in lesser mobility for tokens and that the status information in reality gives an even worse appearance on free resources in the network than for short transfers. In this case a request is rejected, if we do not believe that we will find any resources. This mechanism was introduced to avoid wasting control capacity when the resources were drained.

The reason for this is that the status messages only describe "global" tokens which cover all the segments on the bus. A global token can be utilized by any node and is the only type of token in a DRM-system without slot reuse. At loads higher than 1.0, a great number of tokens are segmented and the reuse scheme needs them at new requests. The status message mechanism which we are using, which was designed for a system without reuse, is therefore limited in its ability in helping new requests to find free capacity and can in the worst case result in a higher degree of blocking.

5.3.2 Performance of the distributed token server as a function of bus length

The throughput and access delay of the distributed token manager with varying bus lengths from 1 km to 1000 km are presented in Fig.21. A 16 kbyte packet is sent between establishment and tear-down of the channel. The 1 km and 100 km buses give about the same throughput and access delay as the 10 km bus, because the latency introduced by using a 125 us cycle dominate over the time of flight latency in the system. For the 1000 km long bus, we see that the access delay is found to be shorter than in the 1000 km system using a centralized token server (see Fig.19). At low load tokens are found very close and access delay is about the same for all systems independently of bus length. Even at very high load the

access delay is about 1 millisecond shorter than it is in the centralized system in Fig.19

5.3.3 Bursty and client-server traffic conditions

The centralized token manager system will have the same performance almost independently of the traffic as long as processing and signaling are sufficient. To evaluate the distributed system, we decided to use two other traffic generators. First, we use a generator that simulate user requests arriving in bursts: when a reques arrives, a new request is generated to arrive with 90% probability after 200 us. The result is that bursts of requests arrive to nodes and force high temporal locality of source addresses. Second, to generate traffic more similar to client-server behavior we increase the amount of traffic arriving to 5 server nodes, 0, 25, 50, 75, and 99. Even the probability for a server destination is also higher.

In Fig.22 we present throughput and access delay performance of the distributed token server system.

5.3.4 Discussion

It is clear that the distributed implementation have several benefits compared to the centralized token server: The nodes may share the processing load, there is less need for high-performance token servers, redundancy may be higher and access delay may be shorter for low capacity request. It also scales to longer busses. The drawback is higher blocking. It is also clear that the status message

and status table mechanism must be changed to avoid unnecessary blocking when slot reuse is allowed.

6. Future work and conclusions

We have shown that the DTM fast circuit switching protocol performs well in a dual-bus shared media environment. Two slot (token) management schemes are analyzed. The centralized scheme perform closest to the ideal protocol and is relatively simple.

The distributed scheme is more sensitive to user behavior, relies on status information that is broadcast frequently, and needs a defragmentation scheme. The main advantage with the distributed scheme is that it efficiently decouples the access delay from the round-trip time on the bus.

One result is that the status message scheme does not work well with slot reuse and future work is to redesign it. Further work is to evaluate a scheme that combines the distributed and the centralized token manager using a small set of token server nodes.

The conclusions are that channel establishment overhead can be very small, resulting in high utilization even for small (a few kilobytes) transfers. Access delay is found to be a few hundred microseconds even at high load. The slot reuse scheme can increase the throughput performance by a factor of two and can be implemented without introducing any extra hardware in the nodes.

C. Supplement

The network is not limited to a dual bus, but can be realized with all types of structures, e.g. a ring structure with an arbitrary number of nodes. The transmission medium can, except optical fibre, be coaxial cable or another transmission medium with a high bandwidth. In the description the transmission medium will

be referred to as optical fibre. The bandwidth of the DTM dual bus is, in the preferred embodiment, partitioned in 125 µs cycles, which in turn are partitioned in time slots of 64 bits. The present invention is not limited to DTM networks with these values, but can be used in networks with cycles and time slots with arbitrary size.

The increase of performance with time slot reuse is clear: at a dual bus where the source and destination pairs are distributed uniformly the throughput has proved to increase with a factor two. The increase of performance can be even higher in other types of networks; i.e. in a dual ring structure with the source and destination pairs distributed uniformly, the throughput can be increased with a factor 4.

A consequence of the DTM's time slot reallocation mechanism is that it takes longer time to establish channels which need greater bandwidth. This "trade-off" can be justified: the type of traffic demanding low medium access delay is normally less sensitive to the bandwidth allocated for the transferring, and such traffic can therefore be accepted without much involving of the reallocation protocol. For transfers demanding greater bandwidth, the access delay will be higher and the reallocation protocol will almost always be used. However, the broad band transfers will probably be less sensitive to access delays.

The simulations described above have shown that the fast circuit-switched protocol of DTM performs well in a dualbus-divided medium environment. Two time slot management schemes are analysed and both are performing well and can take advantage of time slot reuse. The centralized scheme acts nearest the ideal protocol and is at the same time easier to implement. The distributed system is more sensitive to user performance and must therefore rely upon status information which is sent frequently, and upon the defragmentation mechanism to decrease the number of control messages needed for channel

establishing and timeslot reallocation. By using the distributed model with defragmentation mechanism on long buses, better performance can be achieved than when using the centralized model. A resource management system which combines the centralized and the distributed model is also possible by using a smaller set of token server nodes.

Furthermore, the connection establishing overhead can be very low, which results in a high degree of utilization even for small (a few kbyte) transfers. The access delay is some hundred micro seconds even at high loads. The time slot reuse method can increase the performance with a factor two (on a dual bus) without the adding of additional hardware to the nodes. When the time slot reuse method is used it is even more important to use the defragmentation method because fragmentation can be performed both in time slot and segment way.

D. Further detailed description and comments to the description above

In the description above is specially described a centralized system characterized in that only one home node exists for all tokens and in that free (idle) tokens always return directly to the home node.

In the completely distributed system all nodes are home nodes (server nodes) and tokens are evenly distributed among all the home nodes.

With the defragmentation method tokens can even be moved back to their home nodes after the elapse of a significant time (e.g. when they have been free (idle) a certain time, so called "idle-time" or when they have not been in their home node for a certain time, so called "last home-time"). One uses the work gravitation - low gravitation corresponds to long significant time and high gravitation corresponds to short significant time. The centralized system mentioned above will have an infinite high gravitation (significant time = 0) and the distributed

system without defragmentation mechanism will have gravitation = 0 (significant time = ∞).

It will be pointed out that the resource management scheme is applicable on anything between these special cases. The number of home nodes can vary from 1 to the overall number of nodes. The gravitation can independently vary from 0 to infinity. A uniting of adjacent block tokens, the uniting in segment direction ought to be prioritized before uniting in time slot direction.

We are giving some examples in connection to the accompanying drawings to clarify.

In fig. 23 there is disclosed a token graph with two time slot sensequtive block tokens A and B. Normally no partitioning/uniting is performed because the segments then would be shorter.

In fig. 24 there is disclosed two segment consecutive block tokens C and D. Normally, partitioning of the two block tokens C and F is performed according to the broken line. Thereafter is performed uniting of the two formed block tokens C' and D' to a token C'D' with twice as long segment. This results in total in three new block tokens, C", D", and C'D'.

In fig. 25 there is discussed three block tokens E, F, G. Preferably, F and G are united to FG to increase the segment size.

In fig. 26 there is disclosed two block tokens H and I. The requested capacity corresponds to half of H and all of I and shall be sent from node A, NA to node B, NB. H is chosen for transportation, which is partitioned according to the broken line.

In fig. 27 there is disclosed a token I. The requested capacity corresponds to half of I and shall be sent from node A, NA to node B, NB. A part of I is chosen for transportation, which part is laid in the upper side or as in fig. 27 in the lower side of I. The remaining part must

then be partitioned (the block token is normally rectangular). Preferably, this is performed according to the broken line to preserve as large a segment as possible.

Of course, it does not have to be in this way. In some applications one can contemplate uniting in time slot direction before uniting in segment direction.

Normally, the status table only contains information about time slots which are free over all segments. One can also contemplate that even information about segment can be found in these status tables. It will, however, be much more information to spread to the remaining nodes.

Another mechanism which preferably can be implemented will be described below in connection to fig. 28. If node O, NO, has access to the capacity (b-a) and over all segment, but shall send the capacity (b-a) only to node N, NN, then it has no use of the segments NN and thereover. These can then be sent to NN for future demands. The same way if node N, NN has access to the capacity (time slots) (d-c) and shall send this capacity to node M, NM, then it will send even the free segments NM and thereover to the node M, NM for future demands. The token block (NO-NN)*(d-c) will be sent down to the node O, NO, for eventual future demands. In this connection there is required extra signalling whereby this token block (NO-NN)*(d-c) can be sent with an eventually lower priority.

Description of Drawings

Fig. 1 DTM multiplex format.

Fig. 2 DTM nodes connected in a network structure.

Fig. 3 A multi address group.

Fig. 4 Throughput and access delay for different packet sizes.

Fig. 5 Throughput and delay for strict capacity request with re-request (16 kbyte transfers).

Fig. 6 Throughput and access delay for different user demands (16 kbyfe transfers).

Fig. 7 Network throughput and access delay as a function of bus length (16 kbyte transfers).

Fig. 8 Throughput and access delay in a 20x20 grid.

Fig. 9 A DTM network of dual bus structure.

Fig. 10 DTM 125 µs cycle.

Fig. 11 A token map showing time slot number and segment.

Fig. 12 A time slot-segment map showing reuse of time slots.

Fig. 13 A distributed token server.

Fig. 14 User re-request for increasing of the degree of utilization (16 kB).

Fig. 15 Defragmentation:access-delay as a function of simulated time.

Fig. 16 Theoretical dual bus and DTM throughput.

Fig. 17 A centralized token server.

Fig. 18 A centralized token server and time slot reuse.

Fig. 19 A 1000 km bus with a centralized token server.

Fig. 20 Utilization and access-delay:distributed token server.

Fig. 21 1-1000 km buys with distributed token server.

Fig. 22 Distributed token server.

Fig. 23-28 Time slot-segment maps.

Claims

1. A method for centralized management of the capacity in a circuit-switched network with a bus or ring structure, which network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized in**
- that a first node, called server node, being allocated tokens corresponding to all data time slots which are flowing in one direction in a bus or a ring,
- that a second node is requesting tokens corresponding to a certain capacity from the server node, and
- that the server node makes reservations for and transfers tokens corresponding to requested capacity to the other node in event the server node has requested capacity unutilized.

2. A method according to claim 1, **characterized in** that the method is implemented in a network of DTM-type (Dynamic Synchronous Transfer Mode).

3. A method according to claim 1 or 2, **characterized in** that transferred tokens are retransmitted to the server node and released when corresponding data time slots no longer are used for transferring of data.

4. A method according to claim 1 or 2, **characterized in** that transferred tokens are retransmitted to the server node and released when corresponding data time slots have not been used for transferring of data of the other node during a significant time.

5. A method according to any one of claims 1-4, **characterized in** that the server node is chosen as the node which has 1/3 of the nodes situated upstream and 2/3 of the nodes situated downstream.

6. A method for distributed management of the capacity in a circuit-switched network with a bus or ring

structure, wherein the bandwidth is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token (write access), **characterized in**

- that at least two nodes, called server nodes, are defined among which are distributed tokens corresponding to all data time slots which are flowing in one direction in a bus or a ring,
- that a node is requesting tokens corresponding to a certain capacity from at least one of the server nodes, and
- that this server node makes reservations for and transfers tokens corresponding to requested capacity to the requesting node in the event the server node has requested capacity unutilized.

7. A method according to claim 6, **characterized in** that the method is implemented in a network of DTM-type (Dynamic Synchronous Transfer mode).

8. A method according to claim 6 or 7, **characterized in** that several server nodes are asked, make reservations for and transfer tokens, which together correspond to the completely requested capacity, to the requesting node in the event that the server nodes together have the requested capacity unutilized.

9. A method according to any one of claims 6-8, **characterized in** that received tokens are used for establishing of channels by sending of a channel establishing message to the node or the nodes which are designated receiver of data in event that a node has received tokens corresponding to the requested capacity.

10. A method according to any one of claims 6-8, **characterized in** that one or several server nodes makes reservations for and transfers tokens corresponding to all of the unutilized capacity to the requesting node in event

that the server nodes together do not have the requested capacity unutilized.

11. A method according to any one of claims 6-8, **characterized in** that one or several server nodes are not transferring any token to the requesting node in event that the server nodes together do not have the requested capacity unutilized.

12. A method according to claim 10 or 11, **characterized in** that further tokens are requested by a node in event that possibly received tokens do not correspond to the requested capacity.

13. A method according to claim 10 or 11, **characterized in** that tokens received in a node are released in event that these tokens do not correspond to the requested capacity.

14. A method according to claim 10 or 11, **characterized in** that received tokens are used for establishing of channels by sending of a channel establishing message to the node or the nodes which are designated receivers of data in event that a node has not received tokens corresponding to the requested capacity.

15. A method according to any of claims 6-14, **characterized in** that each server node regularly is sending information regarding its idle capacity to the rest of the nodes and each node stores in a status table information received from the other nodes.

16. A method according to any one of claims 6-15, **characterized in** that a node is requesting tokens from the server node which for the moment has the most unutilized capacity.

17. A method according to any one of claims 6-15, **characterized in** that a node is requesting tokens from the server node which is closest and for the moment has the requested capacity unutilized.

18. A method according to claim 16 or 17, **characterized in** that the server node from which capacity is requested will be found by referring to the status table.

19. A method according to any one of claims 6-18, **characterized in** that all nodes in a bus or a ring are defined as server nodes and are allocated at least one token.

20. A method according to claim 19, **characterized in** that all nodes in a bus or a ring are allocated the same number of tokens.

21. A method according to any one of claims 6-20, **characterized in** that transferred tokens are retransmitted to the respective server node, even called home node, and are released when corresponding data time slots no longer are used for data transfer.

22. A method according to any one of claims 6-20, **characterized in** that transferred tokens are released when corresponding data time slots no longer are used for data transfer.

23. A method according to claim 22, **characterized in** that transferred tokens are made reservations for, are retransmitted to the respective server node, even called home node, and are released when corresponding data time slots have not been used for data transferring during a significant time.

24. A method according to any one of claims 1-23, **characterized in** that consecutive tokens in a node are represented by a token, called block token, which is indicated as a number on the first time slot in a row and the total number of time slots in the row.

25. A method according to any one of claims 1-24, **characterized in** that a token is partitioned into at least two tokens corresponding to the same time slot, but to different segments, where a segment refers to a part of a

transmission medium which is connecting at least two nodes.

26. A method according to any one of claims 1-25, **characterized in** that the server node, in the event that it has several time slot- or segment-consecutive groups of tokens to choose among, is making reservations for and transferring the tokens which originate from the smallest group of time slot- or segment-consecutive tokens which satisfies a request of capacity, and wherein the smallest group is defined as the group which product of time slots and segments with predetermined weights is the lowest.

27. A method according to any one of claims 1-26, **characterized in** that groups of tokens, which are at least partly segment-consecutive, are regrouped to maximize the number of consecutive segments for each group of tokens at the expense of the number of consecutive time slots.

28. A node control unit in a node in a circuit-switched network with a bus or ring structure, which network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized in**

- that the node control unit is allocated tokens corresponding to a predetermined number of data time slots which are flowing in one direction in a bus or a ring, and
- that the node control unit is arranged to make reservations for and to transfer tokens to a second node control unit in a second node in event that it receives a request for tokens from the second node control unit and that it has requested capacity unutilized.

29. A node control unit according to claim 28, **characterized in** that the network is of DTM-type (Dynamic Synchronous Transfer Mode).

30. A node, even called server node, in a circuit-switched network with a bus or ring structure, which

network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized by** a node control unit, which is allocated tokens corresponding to a predetermined number of data time slots which are flowing in one direction in a bus or a ring, and is arranged to make reservations for and to transfer tokens to a second node control unit in a second node in event that it receives a request for tokens from the second node control unit and that it has requested capacity unutilized.

31. A node according to claim 30, **characterized in** that the network is of DTM-type (Dynamic Synchronous Transfer Mode).

32. A node according to claim 30 or 31, **characterized in** that the node is allocated tokens corresponding to all of the data time slots which are flowing in one direction in a bus or a ring.

33. A node according to claim 32, **characterized in** that the node has 1/3 of the nodes of the bus or the ring upstream and 2/3 of the nodes of the bus or the ring downstream.

34. A circuit-switched network of a bus or ring structure, which network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized by** a node, even called server node, being allocated tokens corresponding to a predetermined number of data time slots which are flowing in one direction in a bus or a ring, and is arranged to make reservations for and to transfer tokens to a second node in event that it receives a request for tokens from the second node and that it has requested capacity unutilized.

35. A circuit-switched network of a bus or ring structure, which network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized in** that at least two nodes, called server nodes, are defined among which tokens corresponding to all data time slots which are flowing in one direction in a bus or a ring are distributed and in that the server nodes are arranged to make reservations for and transfer tokens to a third node in event that the server nodes receive a request for tokens from the third node and that they have the requested capacity unutilized.

36. A circuit-switched network according to claim 34 or 35, **characterized in** that the network is a network of DTM-type (Dynamic Synchronous Transfer Mode).

37. A circuit-switched network according to any one of claims 34-36, **characterized in** that a node is arranged to use received tokens for establishing of channels by sending a channel establishing message to the node or the nodes which are intended to be receiver of data in event that the node has received tokens corresponding to the requested capacity.

38. A circuit-switched network according to any one of claims 35-37, **characterized in** that one or several server nodes are arranged to make reservations for and to transfer tokens corresponding to all unutilized capacity to the requesting node in event that the server nodes together do not have the requested capacity unutilized.

39. A circuit-switched network according to any one of claims 35-37, **characterized in** that one or several server nodes are arranged to not transfer any token to the requesting node in event that the server nodes together have not the requested capacity unutilized.

40. A circuit-switched network according to any one of claims 35-39, **characterized in** that a node is arranged to use a status table, a relatively regularly updated list of the unutilized capacity of all the server nodes, for decision upon which server node to request tokens from.

AMENDED CLAIMS

[received by the International Bureau on 28 May 1997 (28.05.97); original claims 1-40 replaced by amended claims 1-39 (6 pages)]

CLAIMS

1. A method for centralized management of the capacity in a circuit-switched network with a bus or ring structure, which network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized in**

- that a first node, called server node, being allocated tokens corresponding to all data time slots which are flowing in one direction in a bus or a ring,

- that a second node is requesting tokens corresponding to a certain capacity from the server node,

- that the server node makes reservations for and transfers tokens corresponding to requested capacity to the other node in event the server node has requested capacity unutilized, and

- that transferred tokens are retransmitted to the server node and released when corresponding data time slots have not been used for transferring of data of the other node during a significant time.

2. A method according to claim 1, **characterized in** that the method is implemented in a network of DTM-type (Dynamic Synchronous Transfer Mode).

3. A method according to claim 1 or 2, **characterized in** that transferred tokens are retransmitted to the server node and released when corresponding data time slots no longer are used for transferring of data.

4. A method according to any one of claims 1-3, **characterized in** that the server node is chosen as the node which has 1/3 of the nodes situated upstream and 2/3 of the nodes situated downstream.

5. A method for distributed management of the capacity in a circuit-switched network with a bus or ring structure, wherein the bandwidth is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token (write access), **characterized in**

- that at least two nodes, called server nodes, are defined among which are distributed tokens corresponding to all data time slots which are flowing in one direction in a bus or a ring,

- that a node is requesting tokens corresponding to a certain capacity from at least one of the server nodes, and

- that this server node makes reservations for and transfers tokens corresponding to requested capacity to the requesting node in the event the server node has requested capacity unutilized.

6. A method according to claim 5, **characterized in** that the method is implemented in a network of DTM-type (Dynamic Synchronous Transfer mode).

7. A method according to claim 5 or 6, **characterized in** that several server nodes are asked, make reservations for and transfer tokens, which together correspond to the completely requested capacity, to the requesting node in the event that the server nodes together have the requested capacity unutilized.

8. A method according to any one of claims 5-7, **characterized in** that received tokens are used for establishing of channels by sending of a channel establishing message to the node or the nodes which are designated receiver of data in event that a node has received tokens corresponding to the requested capacity.

9. A method according to any one of claims 5-7, **characterized in** that one or several server nodes makes reservations for and transfers tokens corresponding to all of the unutilized capacity to the requesting node in event that the server nodes together do not have the requested capacity unutilized.

10. A method according to any one of claims 5-7, **characterized in** that one or several server nodes are not transferring any token to the requesting node in event that the server nodes together do not have the requested capacity unutilized.

11. A method according to claim 9 or 10, **characterized in** that further tokens are requested by a node in event that possibly received tokens do not correspond to the requested capacity.

12. A method according to claim 9 or 10, **characterized in** that tokens received in a node are released in event that these tokens do not correspond to the requested capacity.

13. A method according to claim 9 or 10, **characterized in** that received tokens are used for establishing of channels by sending of a channel establishing message to the node or the nodes which are designated receivers of data in event that a node has not received tokens corresponding to the requested capacity.

14. A method according to any of claims 5-13, **characterized in** that each server node regularly is sending information regarding its idle capacity to the rest of the nodes and each node stores in a status table information received from the other nodes.

15. A method according to any one of claims 5-14, **characterized in** that a node is requesting tokens from the server node which for the moment has the most unutilized capacity.

16. A method according to any one of claims 5-14, **characterized in** that a node is requesting tokens from the server node which is closest and for the moment has the requested capacity unutilized.

17. A method according to claim 15 or 16, **characterized in** that the server node from which capacity is requested will be found by referring to the status table.

18. A method according to any one of claims 5-17, **characterized in** that all nodes in a bus or a ring are defined as server nodes and are allocated at least one token.

19. A method according to claim 18, **characterized in** that all nodes in a bus or a ring are allocated the same number of tokens.

20. A method according to any one of claims 5-19, **characterized in** that transferred tokens are retransmitted to the respective server node, even called home node, and are released when corresponding data time slots no longer are used for data transfer.

21. A method according to any one of claims 5-19, **characterized in** that transferred tokens are released when corresponding data time slots no longer are used for data transfer.

22. A method according to claim 21, **characterized in** that transferred tokens are made reservations for, are retransmitted to the respective server node, even called home node, and are released when corresponding data time slots have not been used for data transferring during a significant time.

23. A method according to any one of claims 1-22, **characterized in** that consecutive tokens in a node are represented by a token, called block token, which is indicated as a number on the first time slot in a row and the total number of time slots in the row.

24. A method according to any one of claims 1-23, **characterized in** that a token is partitioned into at least two tokens corresponding to the same time slot, but to different segments, where a segment refers to a part of a transmission medium which is connecting at least two nodes.

25. A method according to any one of claims 1-24, **characterized in** that the server node, in the event that it has several time slot- or segment-consecutive groups of tokens to choose among, is making reservations for and transferring the tokens which originate from the smallest group of time slot- or segment-consecutive tokens

which satisfies a request of capacity, and wherein the smallest group is defined as the group which product of time slots and segments with predetermined weights is the lowest.

26. A method according to any one of claims 1-25, **characterized in** that groups of tokens, which are at least partly segment-consecutive, are regrouped to maximize the number of consecutive segments for each group of tokens at the expense of the number of consecutive time slots.

27. A node control unit in a node in a circuit-switched network with a bus or ring structure, which network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized in**

- that the node control unit is allocated tokens corresponding to a predetermined number of data time slots which are flowing in one direction in a bus or a ring, and

- that the node control unit is arranged to make reservations for and to transfer tokens to a second node control unit in a second node in event that it receives a request for tokens from the second node control unit and that it has requested capacity unutilized.

28. A node control unit according to claim 27, **characterized in** that the network is of DTM-type (Dynamic Synchronous Transfer Mode).

29. A node, even called server node, in a circuit-switched network with a bus or ring structure, which network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized by** a node control unit, which is allocated tokens corresponding to a predetermined number of data time slots which are flowing in one direction in a bus or a ring, and is arranged to make reservations for and to transfer tokens to a second node control unit in a second node in event that it receives a request for tokens from the second node control unit and that it has requested capacity unutilized.

30. A node according to claim 29, **characterized in** that the network is of DTM-type (Dynamic Synchronous Transfer Mode).

31. A node according to claim 29 or 30, **characterized in** that the node is allocated tokens corresponding to all of the data time slots which are flowing in one direction in a bus or a ring.

32. A node according to claim 31, **characterized in** that the node has 1/3 of the nodes of the bus or the ring upstream and 2/3 of the nodes of the bus or the ring downstream.

33. A circuit-switched network of a bus or ring structure, which network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized by** a node, even called server node, being allocated tokens corresponding to a predetermined number of data time slots which are flowing in one direction in a bus or a ring, and is arranged to make reservations for and to transfer tokens to a second node in event that it receives a request for tokens from the second node and that it has requested capacity unutilized.

34. A circuit-switched network of a bus or ring structure, which network uses a bandwidth which is partitioned in cycles, which in turn are partitioned in control time slots for signalling and data time slots for transfer of data, and wherein each data time slot is associated with a token, **characterized in** that at least two nodes, called server nodes, are defined among which tokens corresponding to all data time slots which are flowing in one direction in a bus or a ring are distributed and in that the server nodes are arranged to make reservations for and transfer tokens to a third node in event that the server nodes receive a request for tokens from the third node and that they have the requested capacity unutilized.

35. A circuit-switched network according to claim 33 or 34, **characterized in** that the network is a network of DTM-type (Dynamic Synchronous Transfer Mode).

36. A circuit-switched network according to any one of claims 33-35, **characterized in** that a node is arranged to use received tokens for establishing of channels by sending a channel establishing message to the node or the nodes which are intended to be receiver of data in event that the node has received tokens corresponding to the requested capacity.

37. A circuit-switched network according to any one of claims 34-36, **characterized in** that one or several server nodes are arranged to make reservations for and to transfer tokens corresponding to all unutilized capacity to the requesting node in event that the

server nodes together do not have the requested capacity unutilized.

38. A circuit-switched network according to any one of claims 34-36, **characterized in** that one or several server nodes are arranged to not transfer any token to the requesting node in event that the server nodes together have not the requested capacity unutilized.

39. A circuit-switched network according to any one of claims 34-38, **characterized in** that a node is arranged to use a status table, a relatively regularly updated list of the unutilized capacity of all the server nodes, for decision upon which server node to request tokens from.

Fig. 1

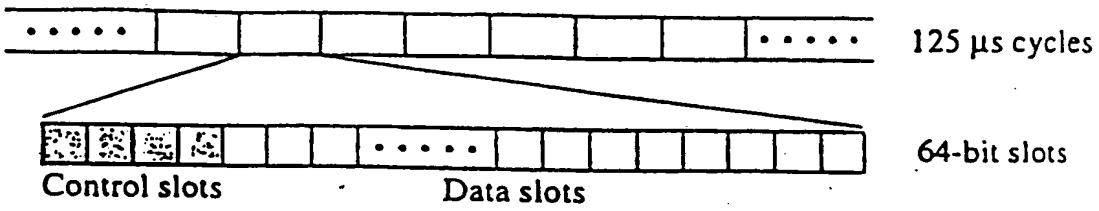


Fig. 2

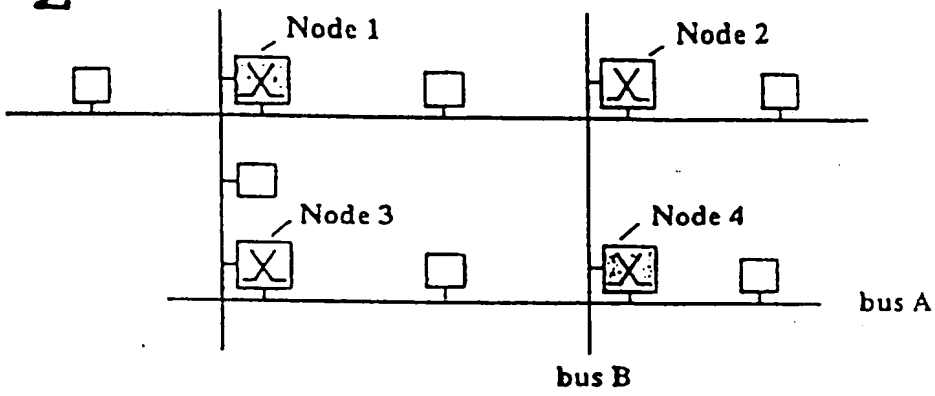


Fig. 3

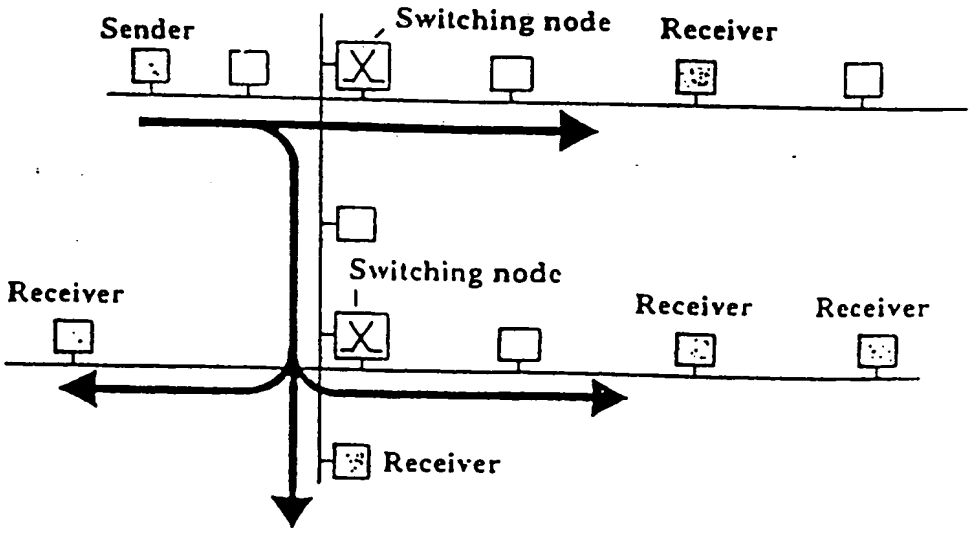
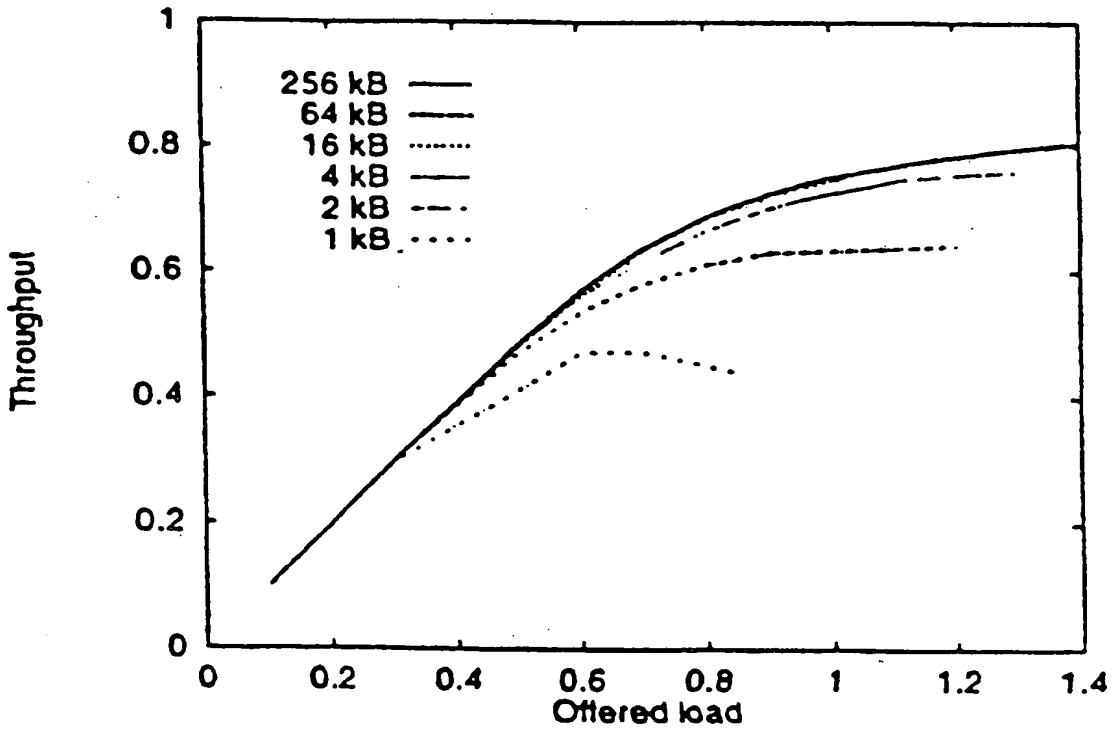


Fig. 4


Throughput
Offered load
256 kB
64 kB
16 kB
4 kB
2 kB
1 kB
0
0.2
0.4
0.6
0.8
1
1.2
1.4

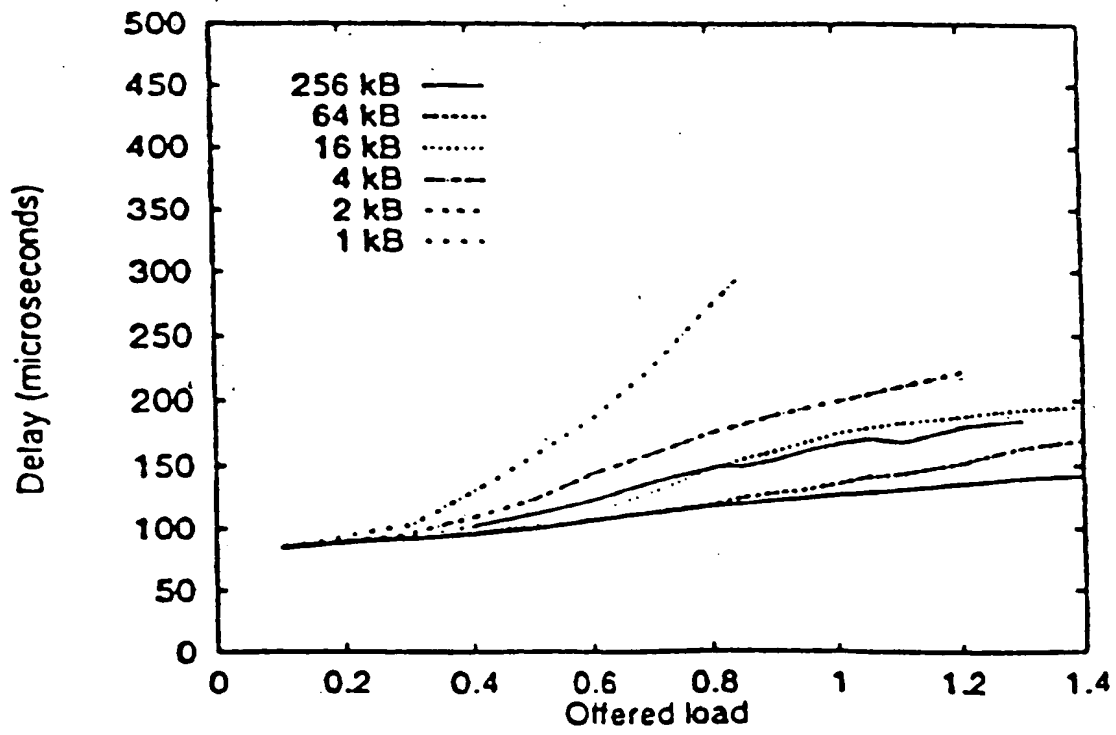
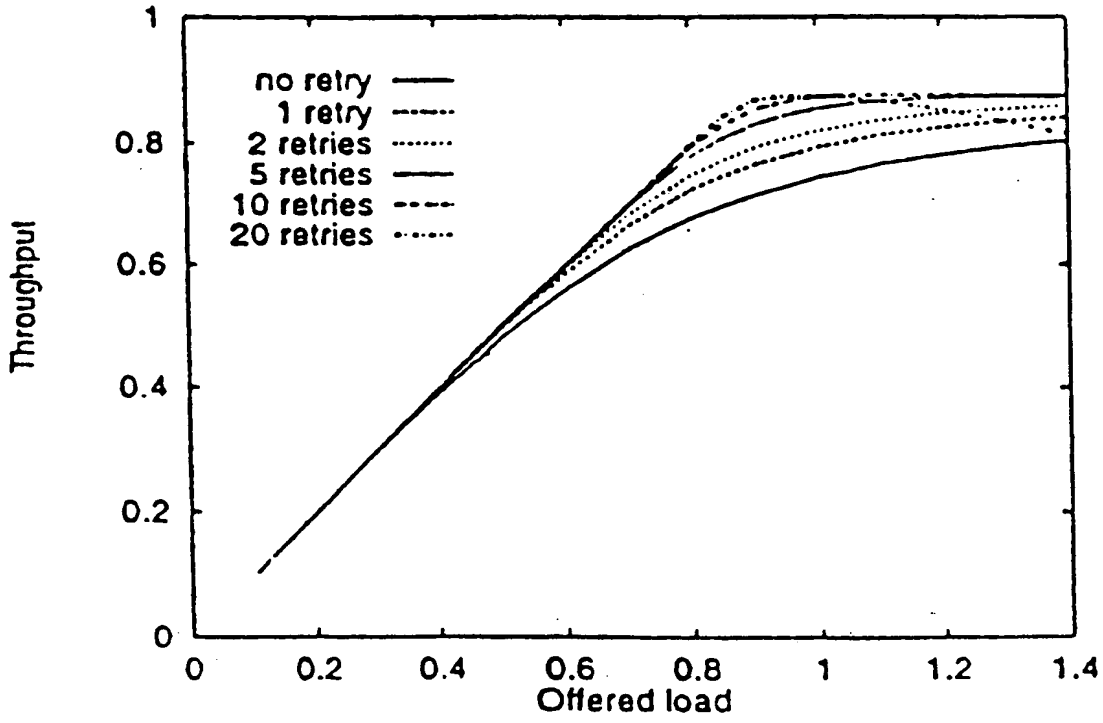

Delay (microseconds)
Offered load
256 kB
64 kB
16 kB
4 kB
2 kB
1 kB
0
50
100
150
200
250
300
350
400
450
500
0.2
0.4
0.6
0.8
1
1.2
1.4

Fig. 5


Throughput
Offered load
no retry
1 retry
2 retries
5 retries
10 retries
20 retries
0
0.2
0.4
0.6
0.8
1
1.2
1.4

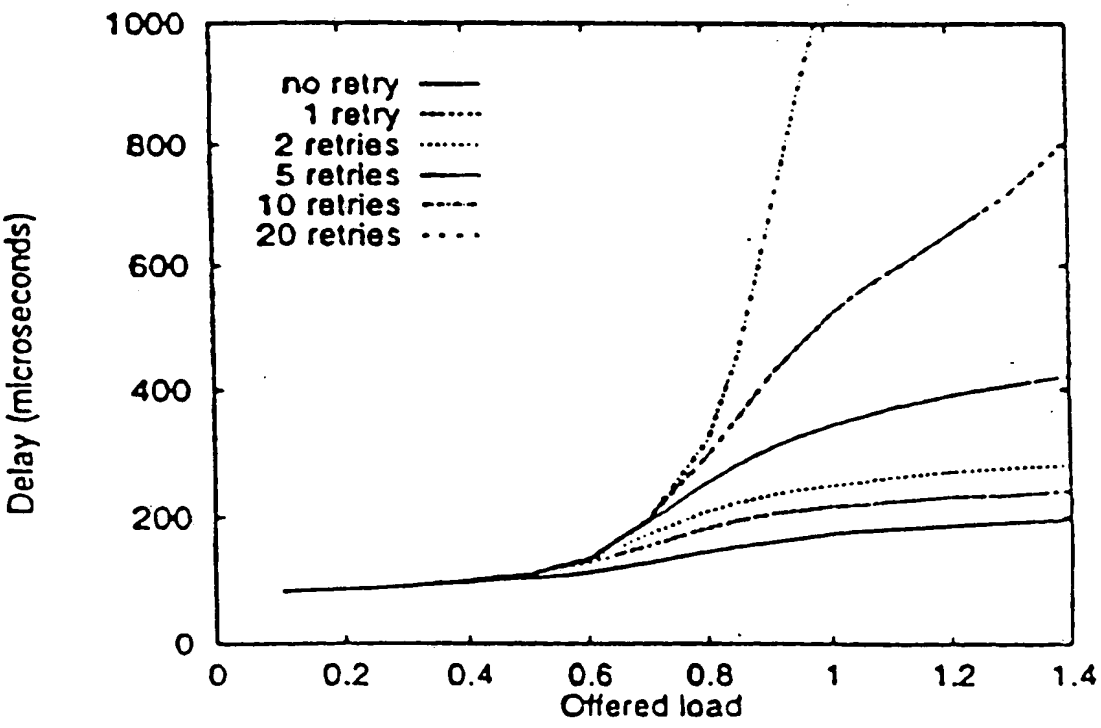
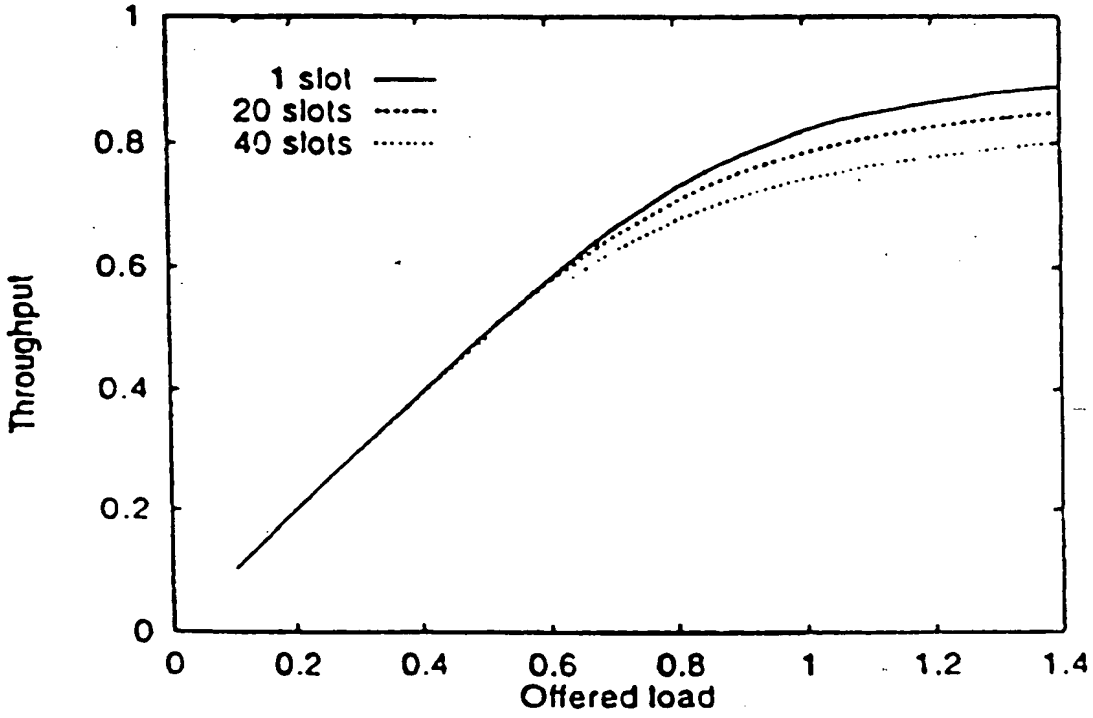

Delay (microseconds)
Offered load
no retry
1 retry
2 retries
5 retries
10 retries
20 retries
0
200
400
600
800
1000
0.2
0.4
0.6
0.8
1
1.2
1.4

Fig. 6


1 slot
20 slots
40 slots
Throughput
Offered load
0
0.2
0.4
0.6
0.8
1
1.2
1.4

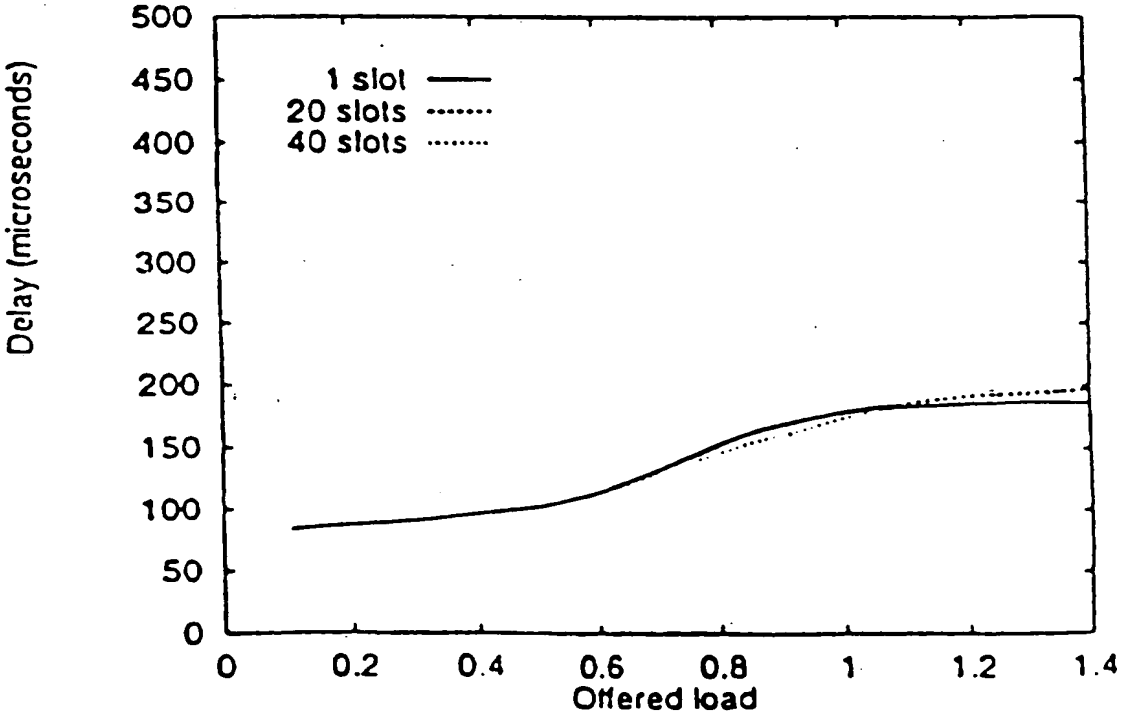
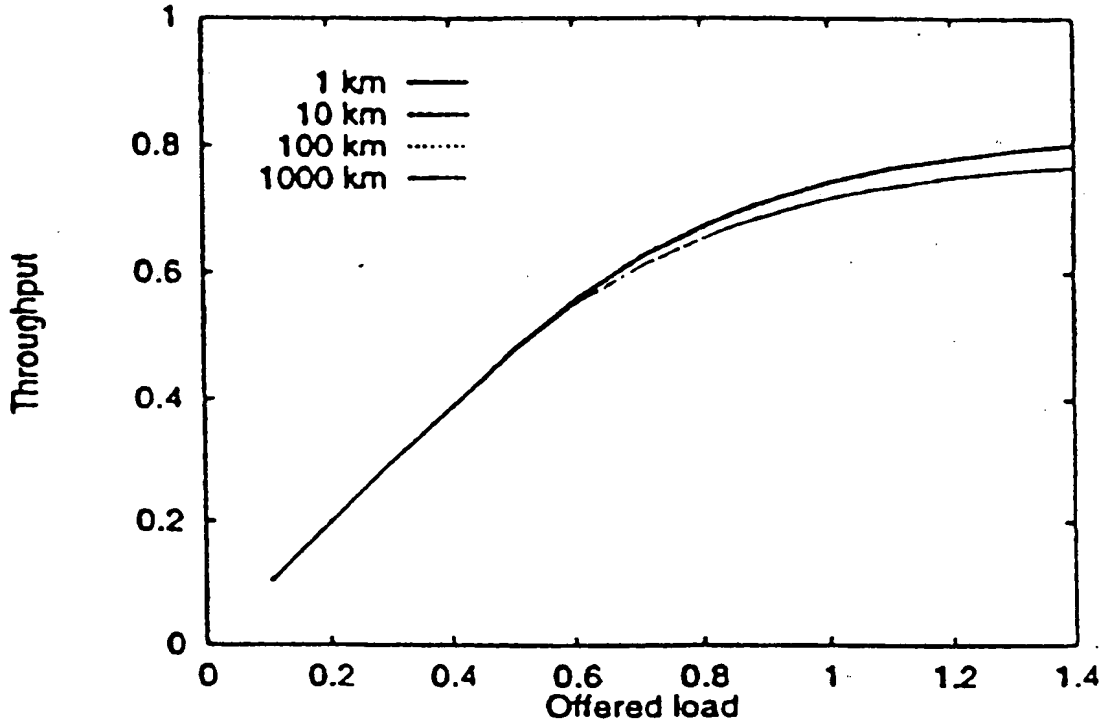

1 slot
20 slots
40 slots
Delay (microseconds)
Offered load
0
50
100
150
200
250
300
350
400
450
500
0.2
0.4
0.6
0.8
1
1.2
1.4

Fig. 7


Throughput
Offered load
1 km
10 km
100 km
1000 km
0
0.2
0.4
0.6
0.8
1
1.2
1.4

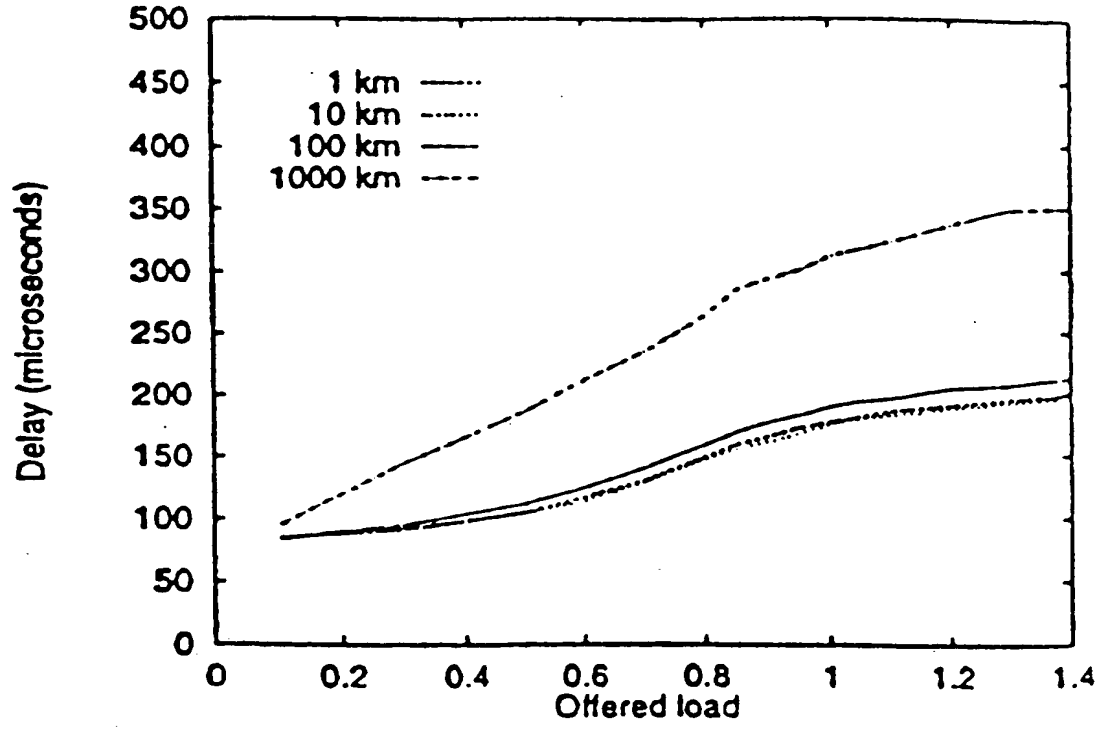
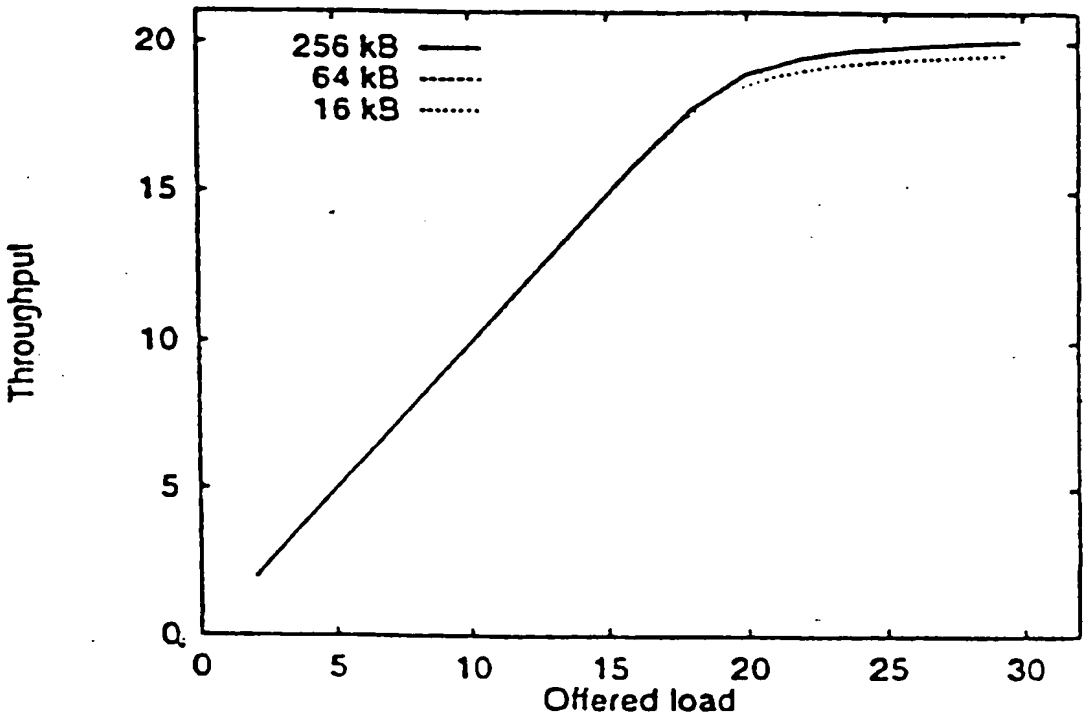

Delay (microseconds)
Offered load
1 km
10 km
100 km
1000 km
0
50
100
150
200
250
300
350
400
450
500
0.2
0.4
0.6
0.8
1
1.2
1.4

Fig. 8


256 kB
64 kB
16 kB
Throughput
Offered load
0
5
10
15
20
25
30

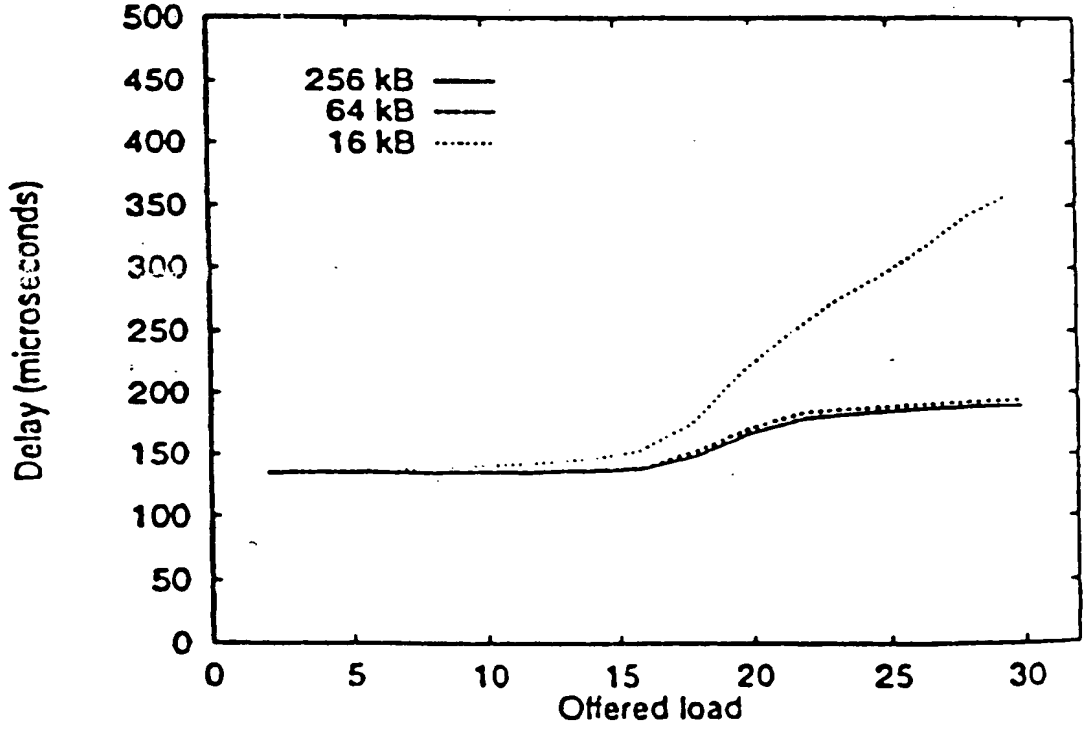

256 kB
64 kB
16 kB
Delay (microseconds)
Offered load
0
50
100
150
200
250
300
350
400
450
500
5
10
15
20
25
30

Fig. 9

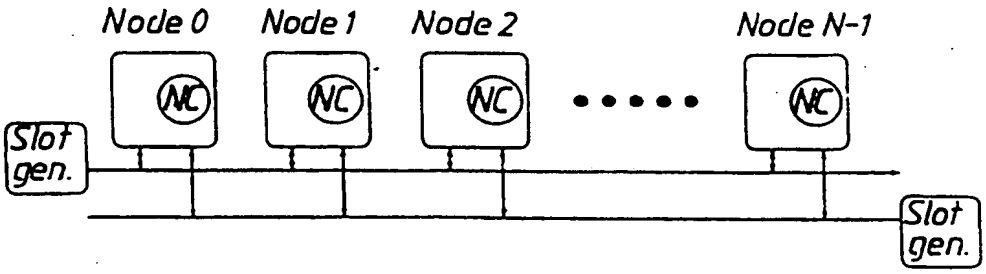

Node 0
Node 1
Node 2
Node N-1
NC
NC
NC
NC
Slot gen.
Slot gen.

Fig. 10

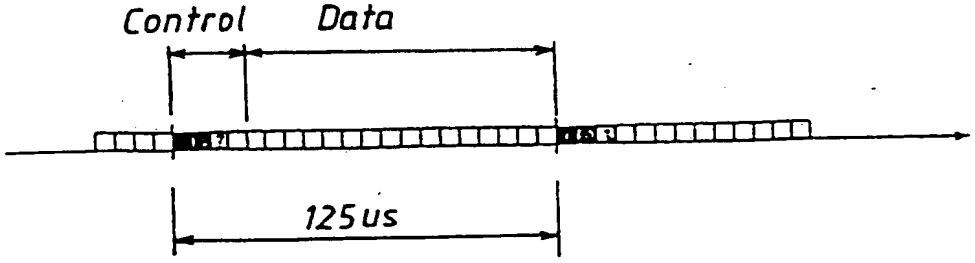

Control
Data
125 us

Fig. 11

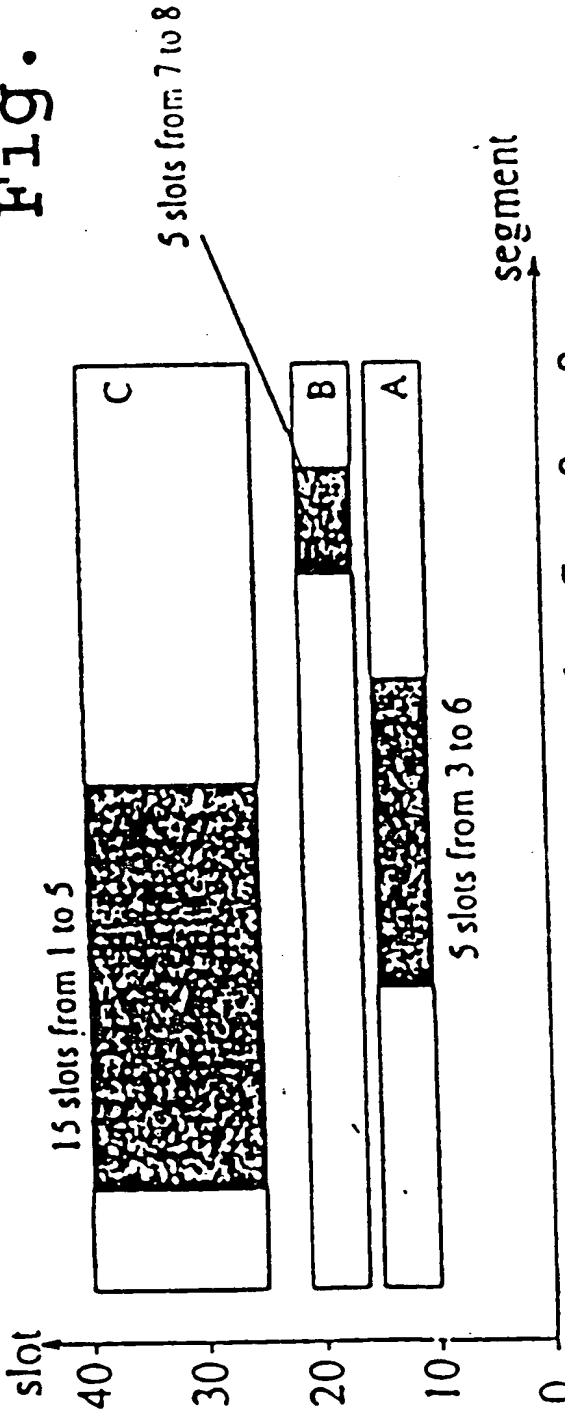

5 slots from 7 to 8
C
B
A
15 slots from 1 to 5
5 slots from 3 to 6
segment
0 1 2 3 4 5 6 7 8 9
slot
0 10 20 30 40

Fig. 12

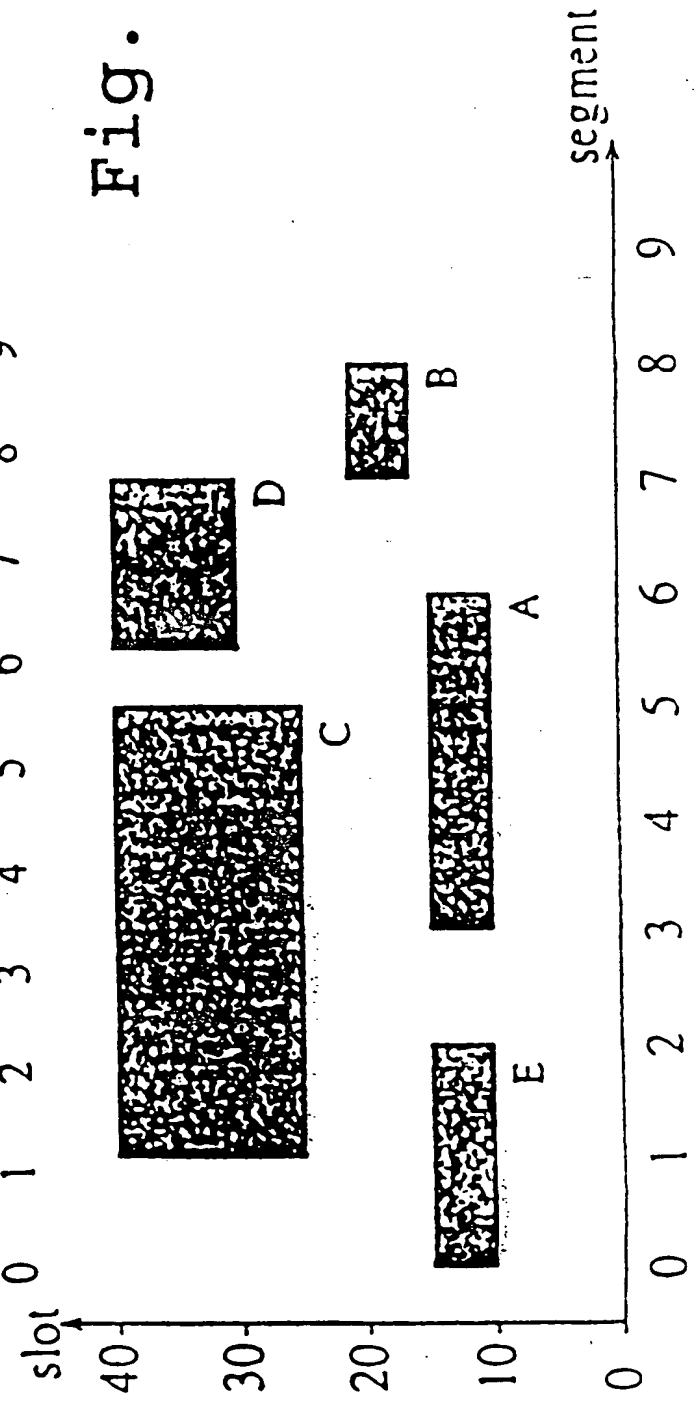
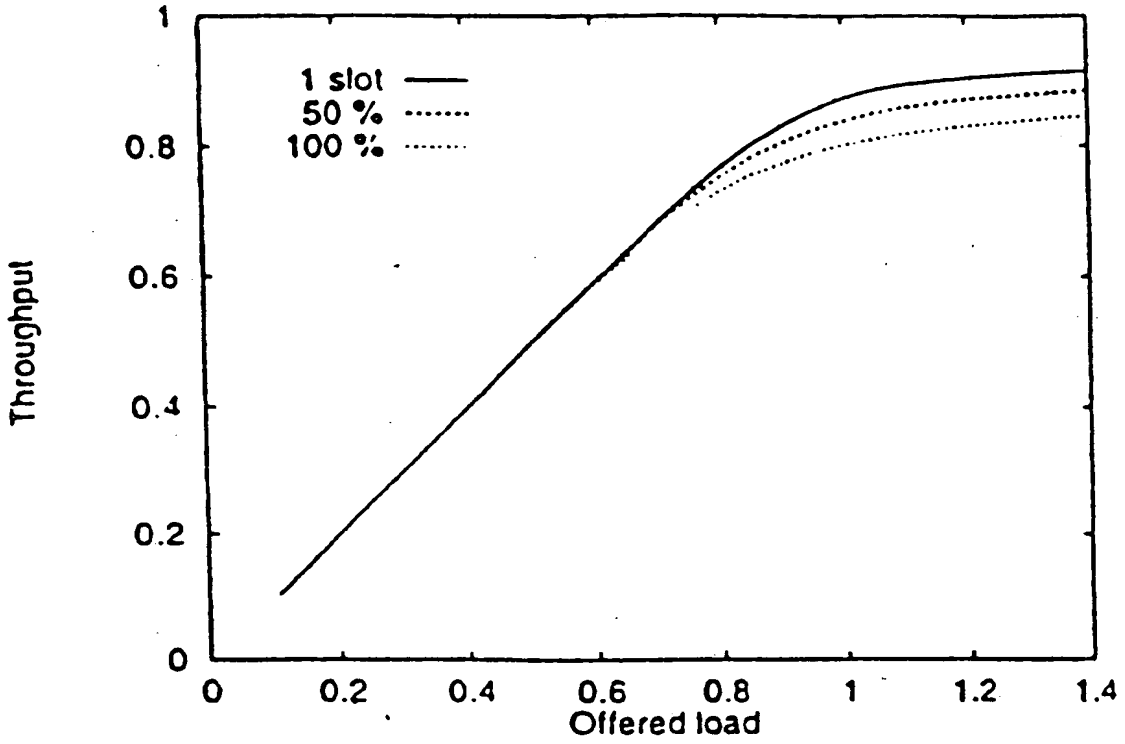

D
B
C
A
E
segment
0 1 2 3 4 5 6 7 8 9
slot
0 10 20 30 40

Fig. 13


1 slot
50 %
100 %
Throughput
Offered load
0
0.2
0.4
0.6
0.8
1
1.2
1.4

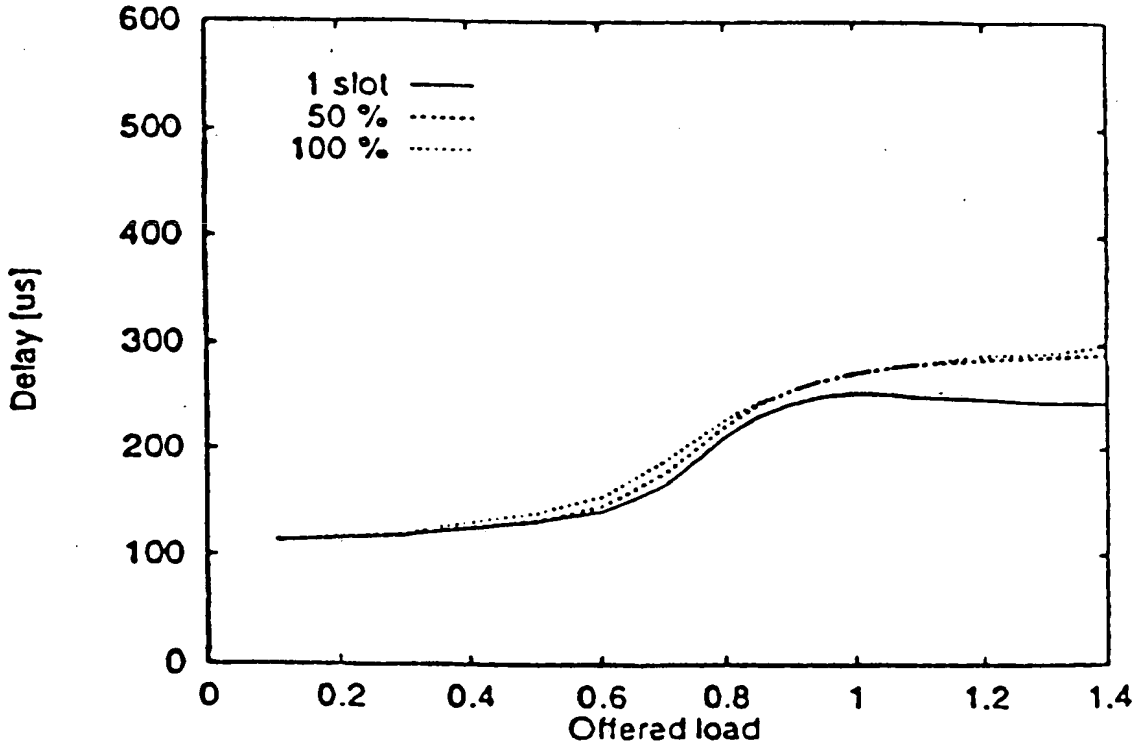
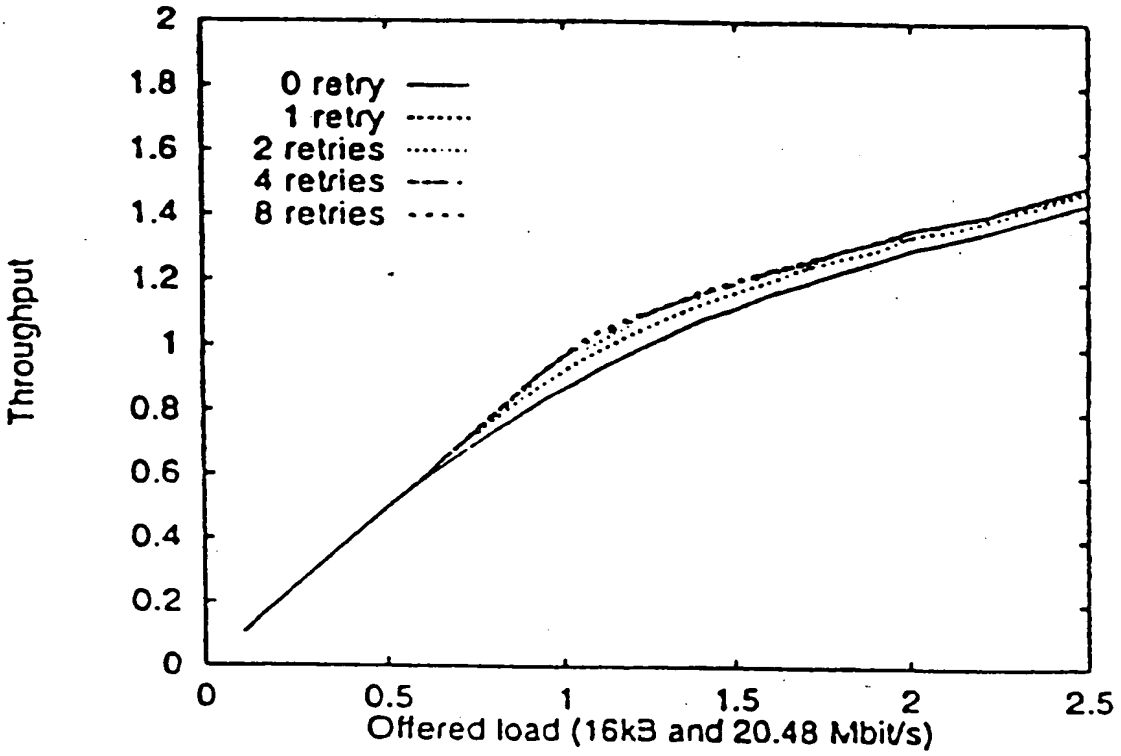

1 slot
50 %
100 %
Delay [us]
Offered load
0
100
200
300
400
500
600
0.2
0.4
0.6
0.8
1
1.2
1.4

Fig. 14


Throughput
Offered load (16kB and 20.48 Mbit/s)
0 retry
1 retry
2 retries
4 retries
8 retries
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
0
0.5
1
1.5
2
2.5

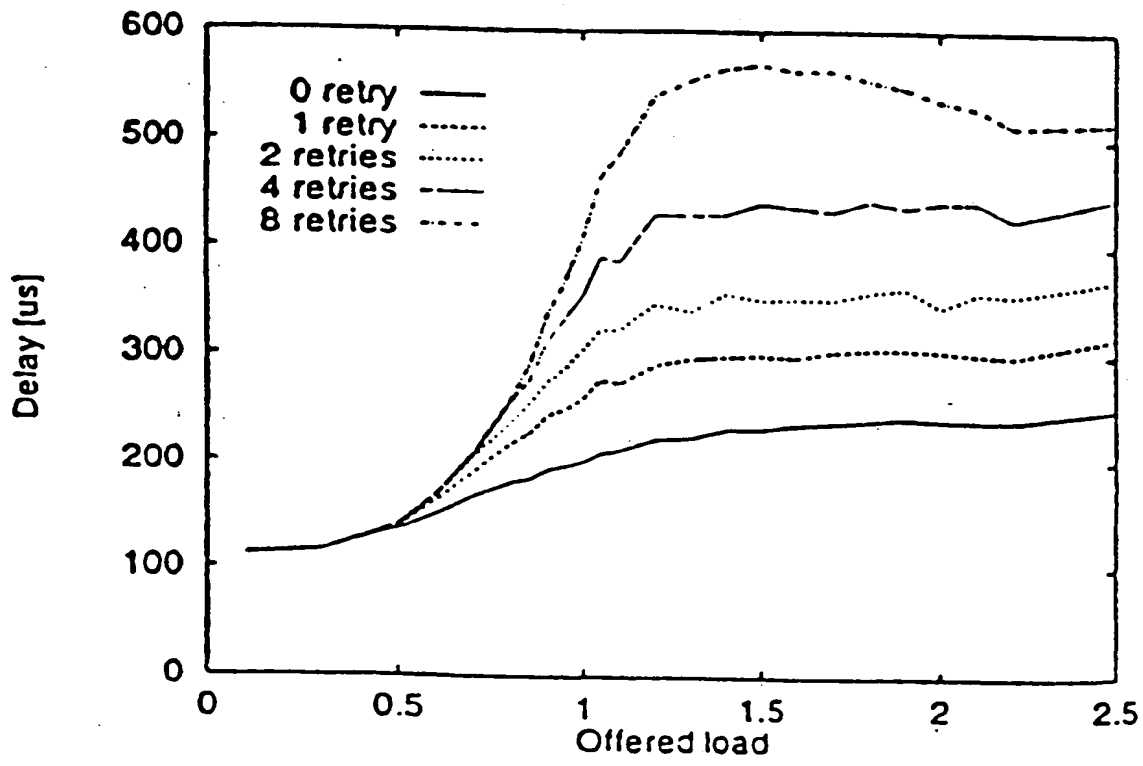

Delay [us]
Offered load
0 retry
1 retry
2 retries
4 retries
8 retries
0
100
200
300
400
500
600
0
0.5
1
1.5
2
2.5

Fig. 15

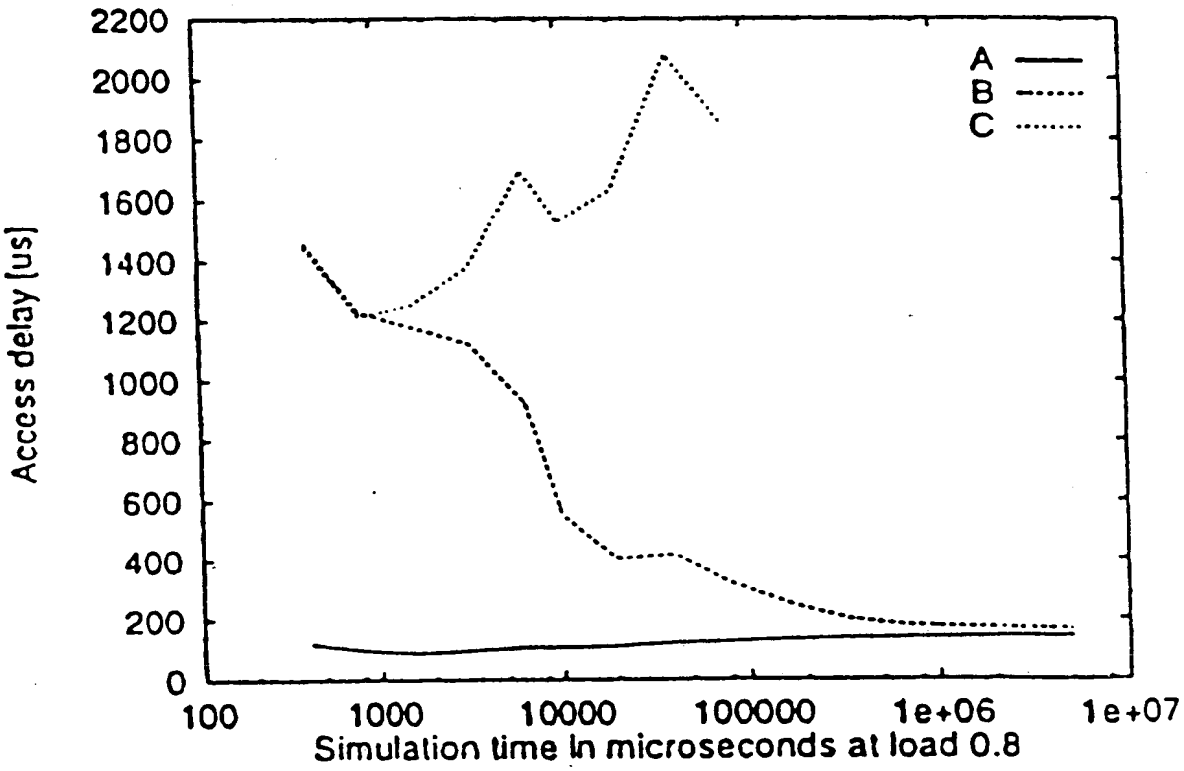
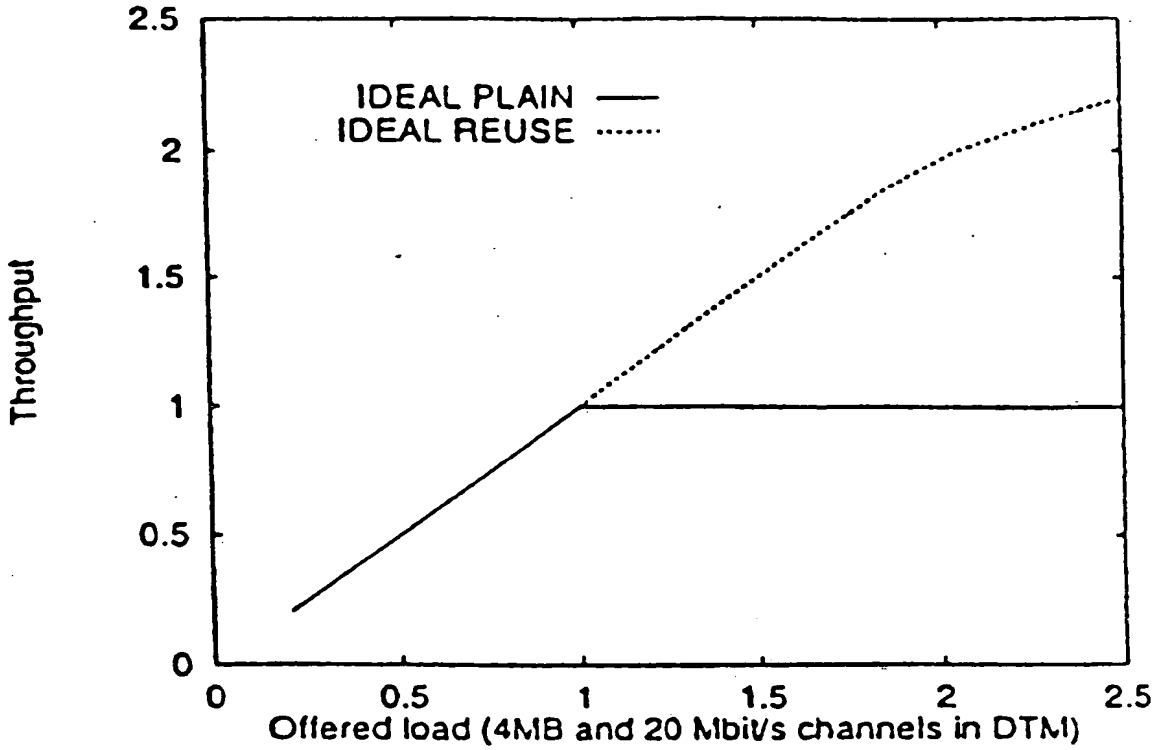

2200
2000
1800
1600
1400
1200
1000
800
600
400
200
0
Access delay [us]
100
1000
10000
100000
1e+06
1e+07
Simulation time in microseconds at load 0.8
A
B
C

Fig. 16


IDEAL PLAIN
IDEAL REUSE
Throughput
Offered load (4MB and 20 Mbit/s channels in DTM)
0
0.5
1
1.5
2
2.5

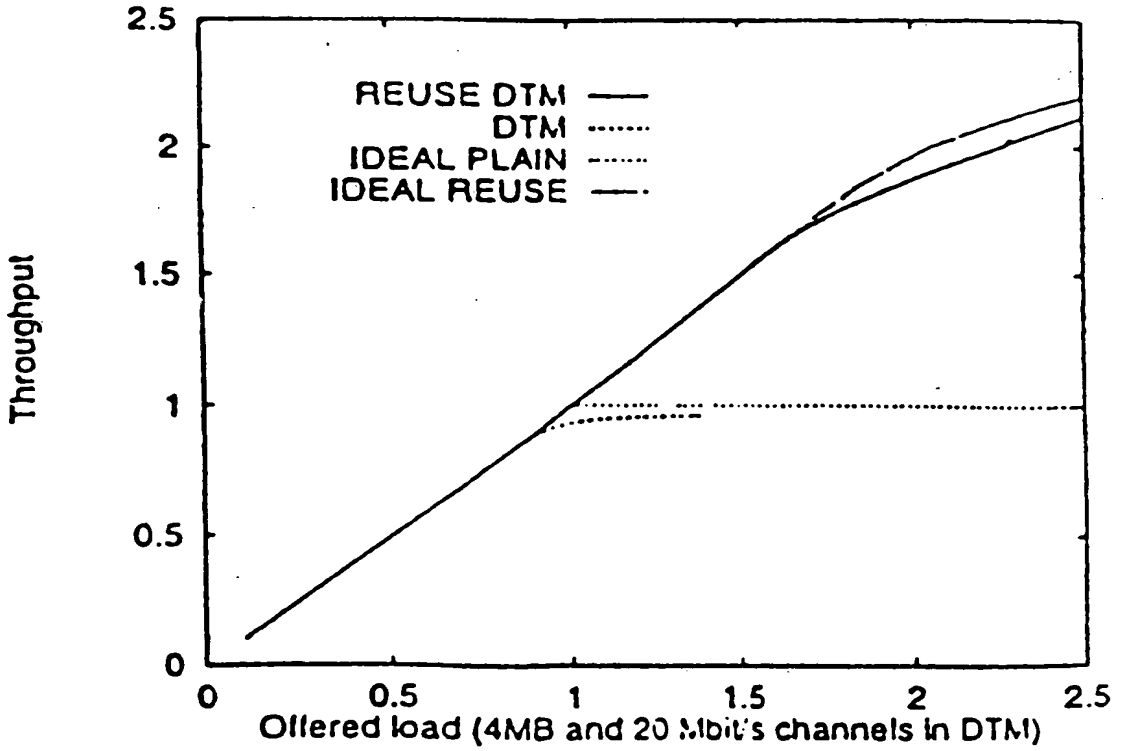
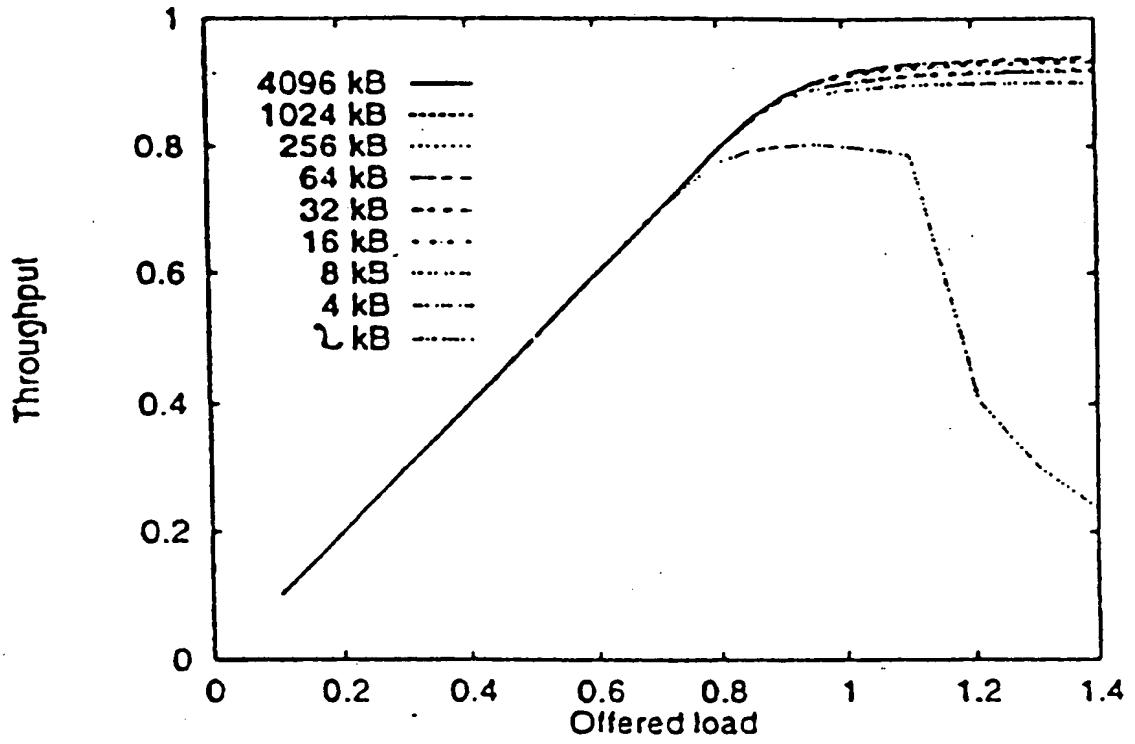

REUSE DTM
DTM
IDEAL PLAIN
IDEAL REUSE
Throughput
Offered load (4MB and 20 Mbit/s channels in DTM)
0
0.5
1
1.5
2
2.5

Fig. 17


Throughput
Offered load
4096 kB
1024 kB
256 kB
64 kB
32 kB
16 kB
8 kB
4 kB
1 kB
0
0.2
0.4
0.6
0.8
1
1.2
1.4

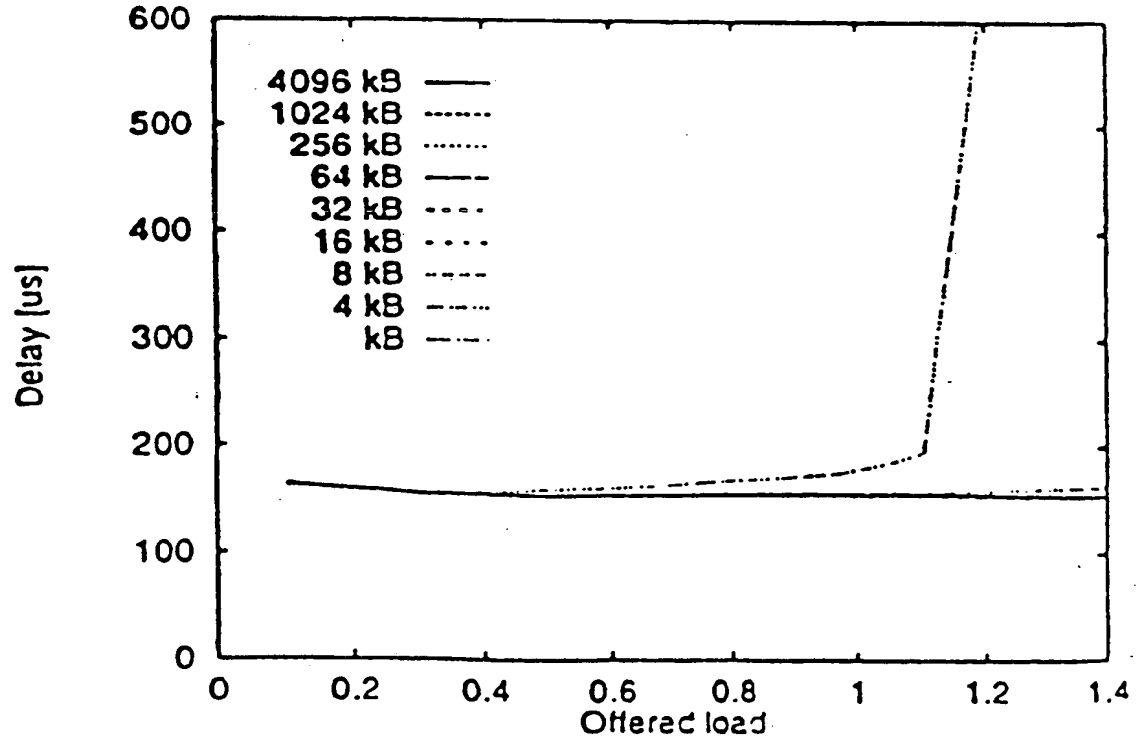
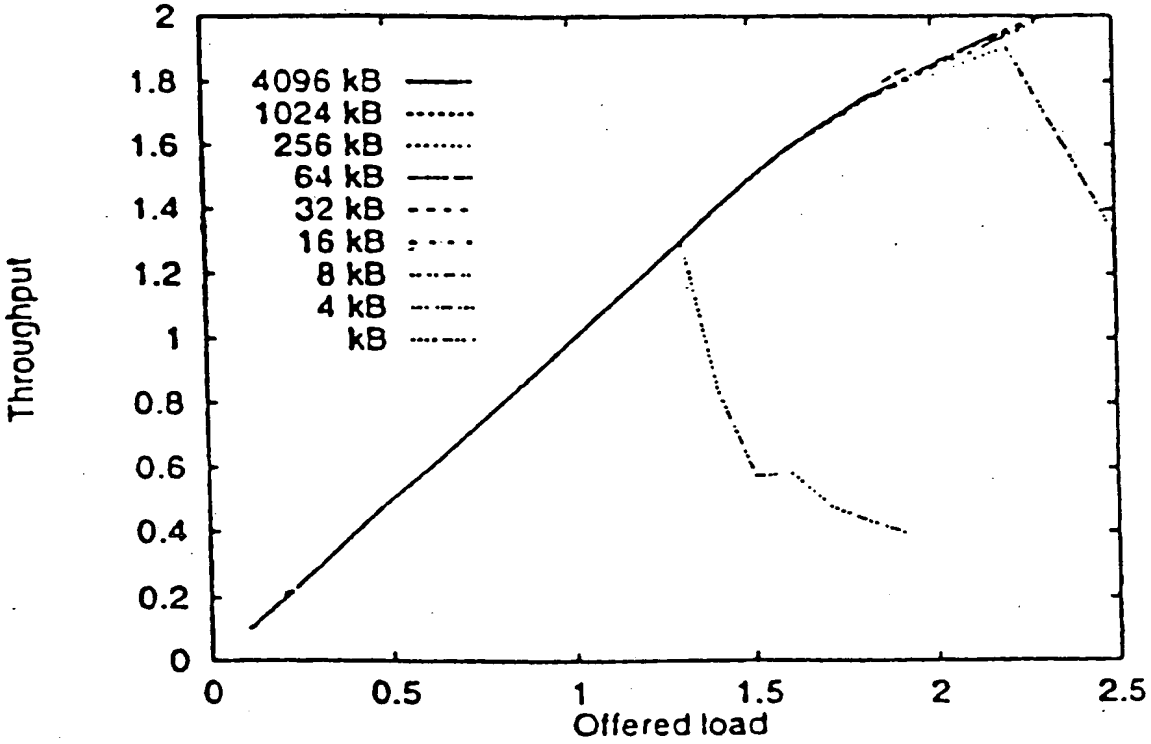

Delay [us]
Offered load
4096 kB
1024 kB
256 kB
64 kB
32 kB
16 kB
8 kB
4 kB
kB
0
100
200
300
400
500
600
0.2
0.4
0.6
0.8
1
1.2
1.4

Fig. 18


Throughput
Offered load
4096 kB
1024 kB
256 kB
64 kB
32 kB
16 kB
8 kB
4 kB
kB
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
0
0.5
1
1.5
2
2.5

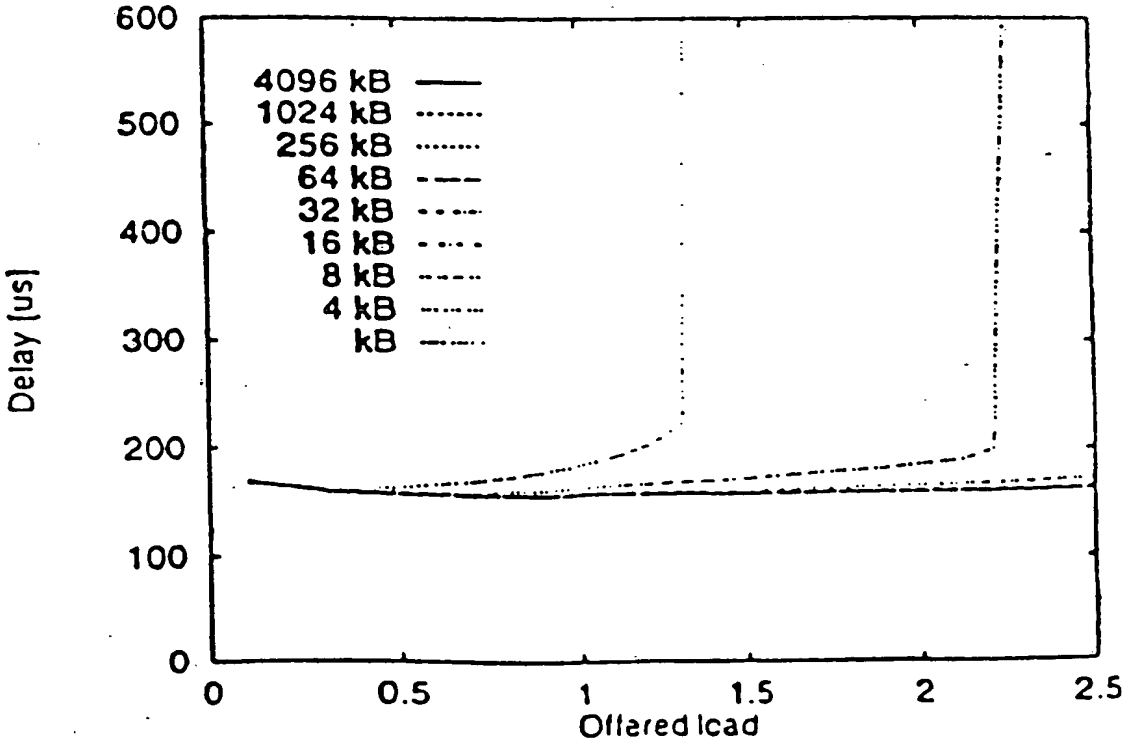
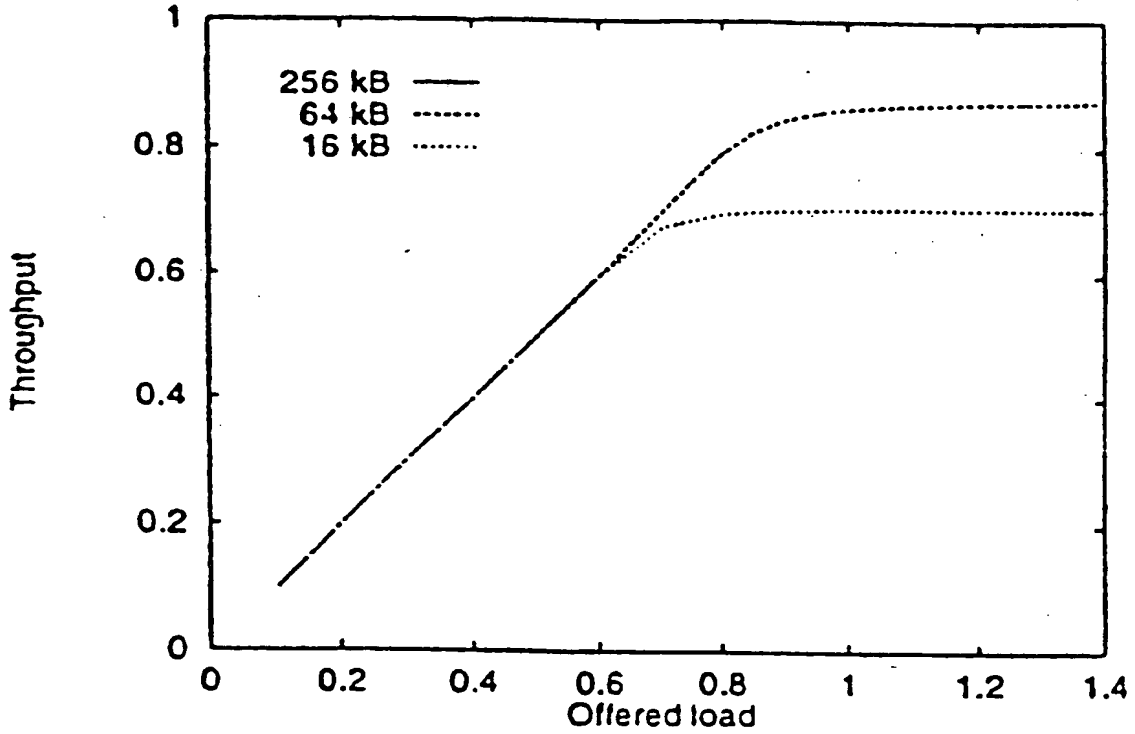

Delay [us]
Offered load
4096 kB
1024 kB
256 kB
64 kB
32 kB
16 kB
8 kB
4 kB
kB
0
100
200
300
400
500
600
0
0.5
1
1.5
2
2.5

Fig. 19


Throughput
Offered load
256 kB
64 kB
16 kB
0
0.2
0.4
0.6
0.8
1
1.2
1.4

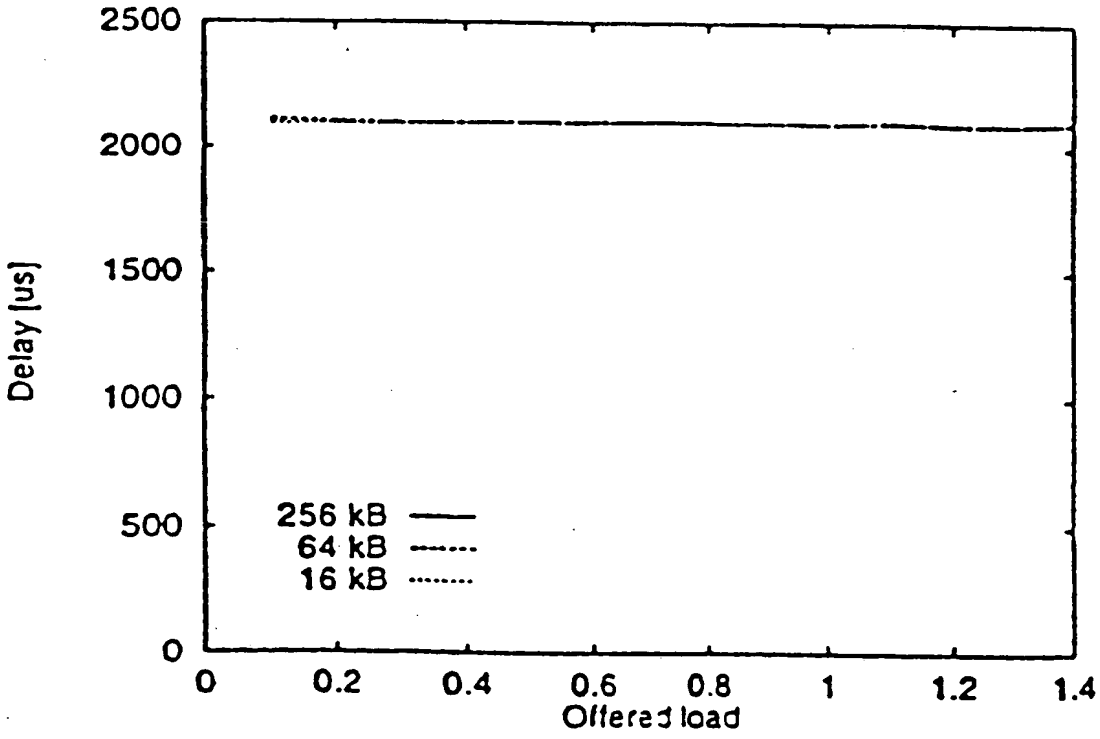
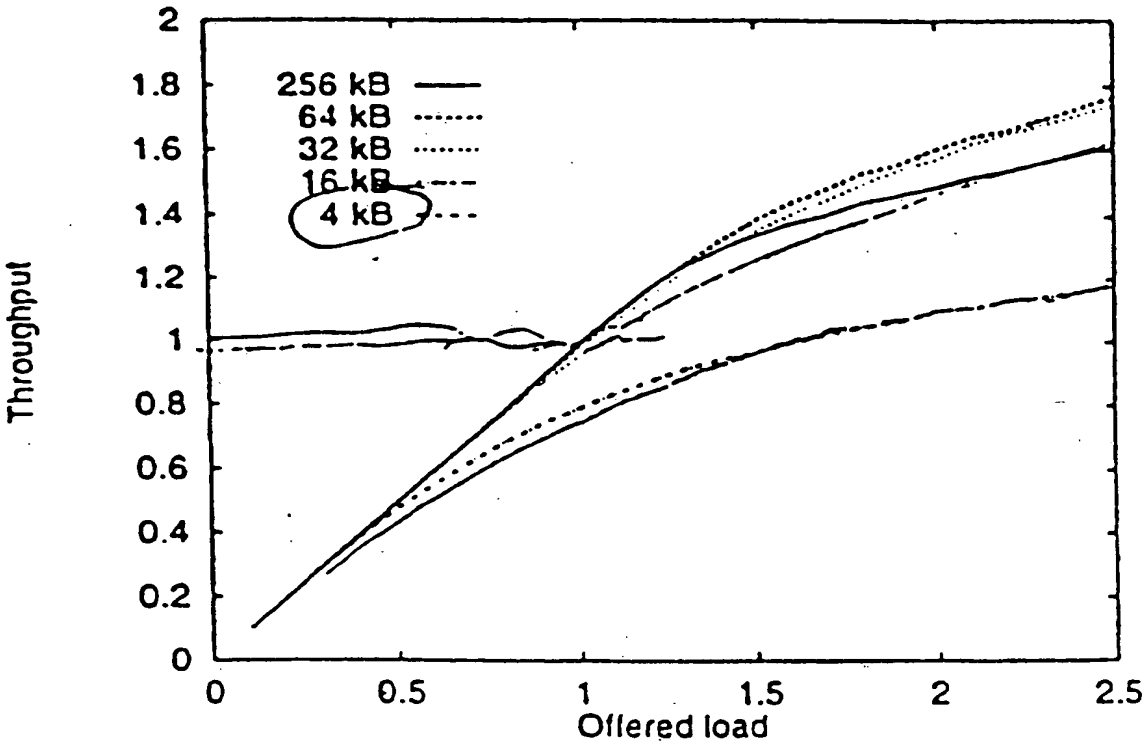

Delay [us]
Offered load
256 kB
64 kB
16 kB
0
500
1000
1500
2000
2500
0.2
0.4
0.6
0.8
1
1.2
1.4

Fig. 20


Throughput
Offered load
256 kB
64 kB
32 kB
16 kB
4 kB
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
0
0.5
1
1.5
2
2.5

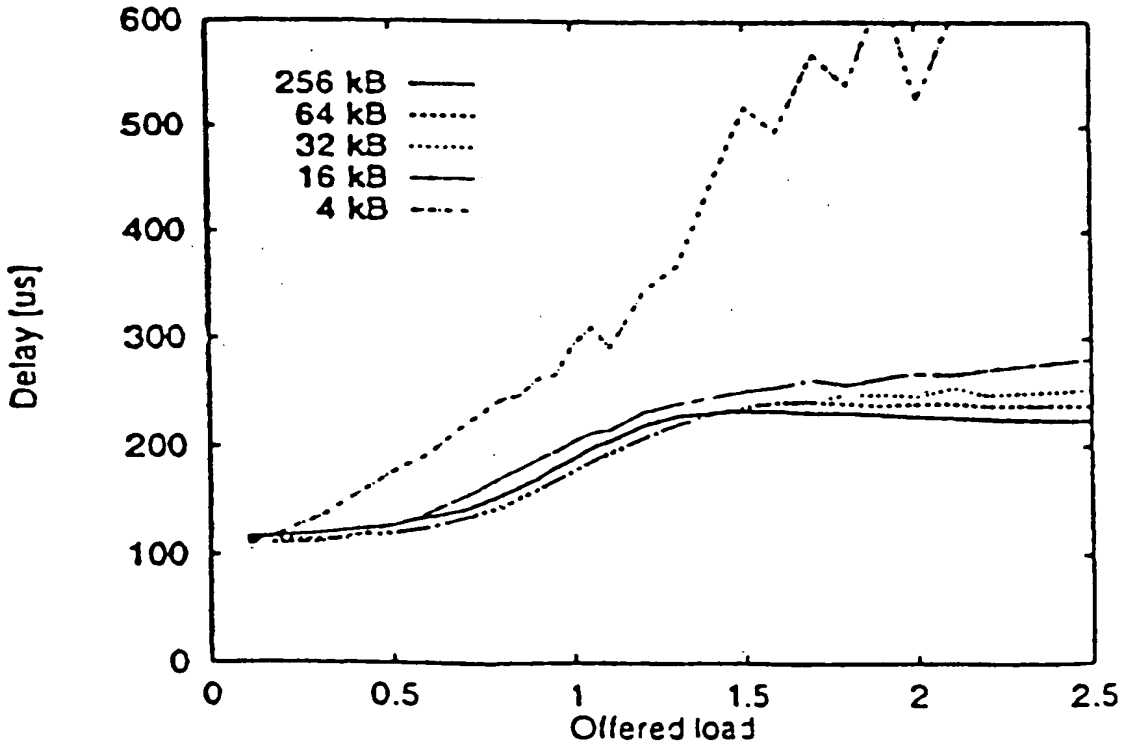
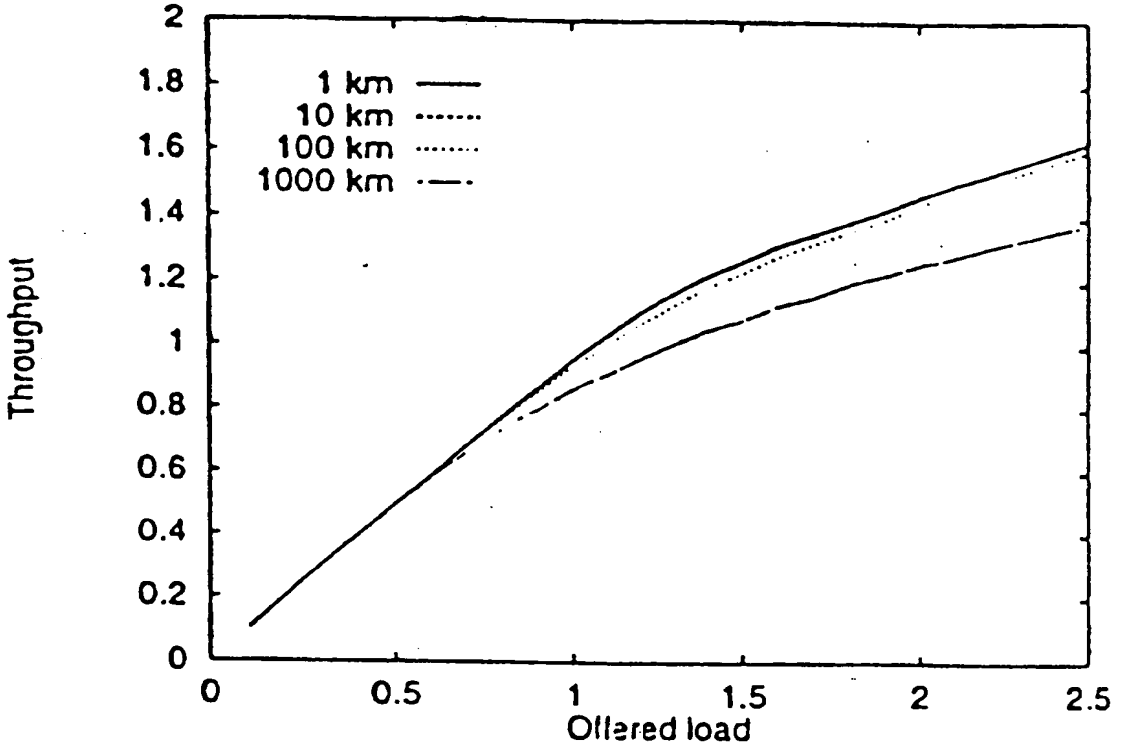

Delay [us]
Offered load
256 kB
64 kB
32 kB
16 kB
4 kB
0
100
200
300
400
500
600
0
0.5
1
1.5
2
2.5

Fig. 21


Throughput
Offered load
1 km
10 km
100 km
1000 km
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
0.5
1.5
2.5

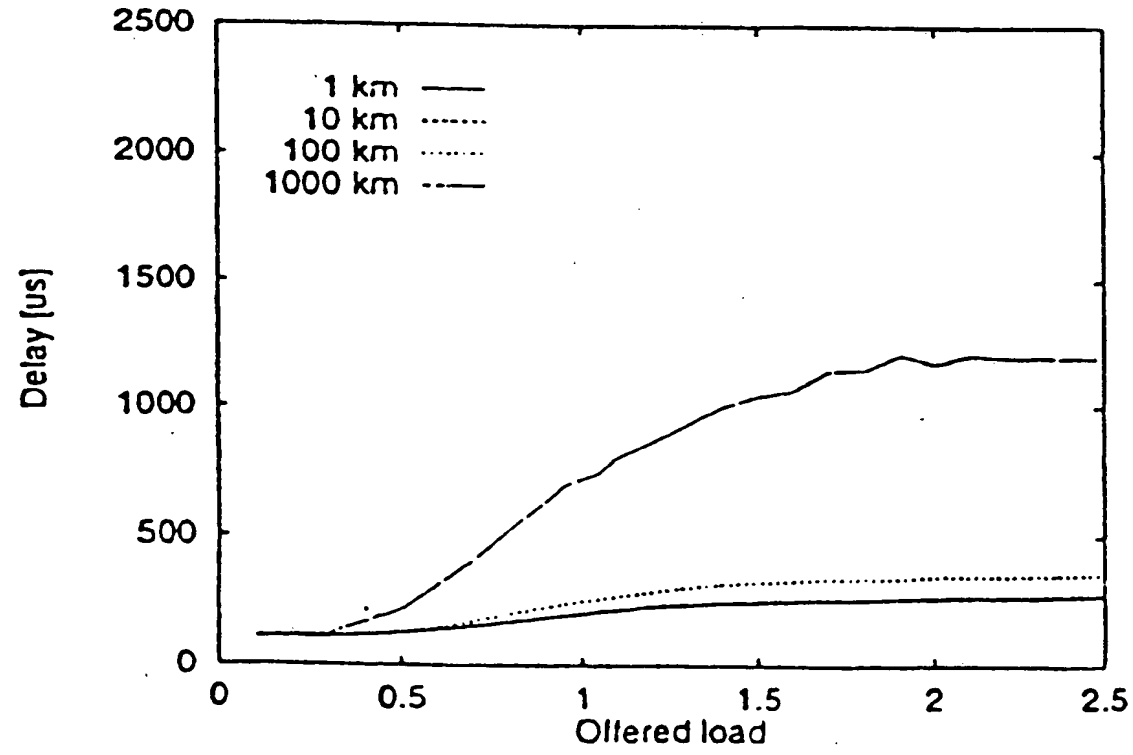
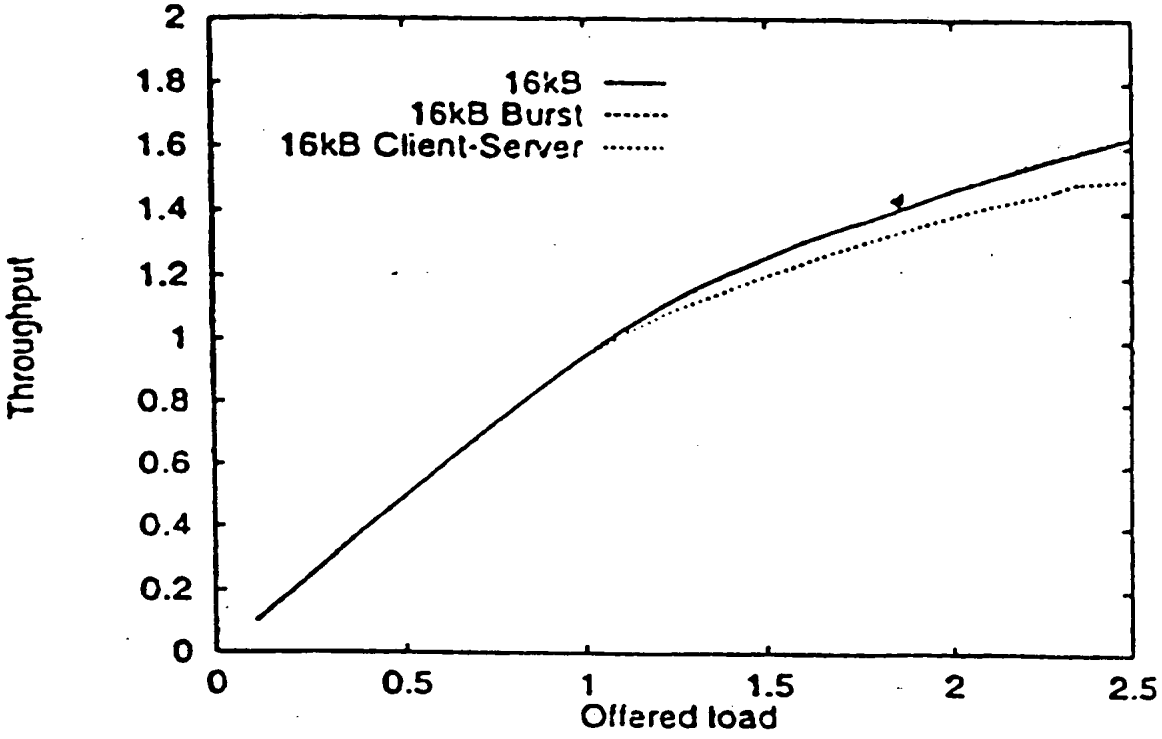

Delay [us]
Offered load
1 km
10 km
100 km
1000 km
0
500
1000
1500
2000
2500
0.5
1
1.5
2
2.5

Fig. 22


16kB
16kB Burst
16kB Client-Server
Throughput
Offered load
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
0.5
1.5
2.5

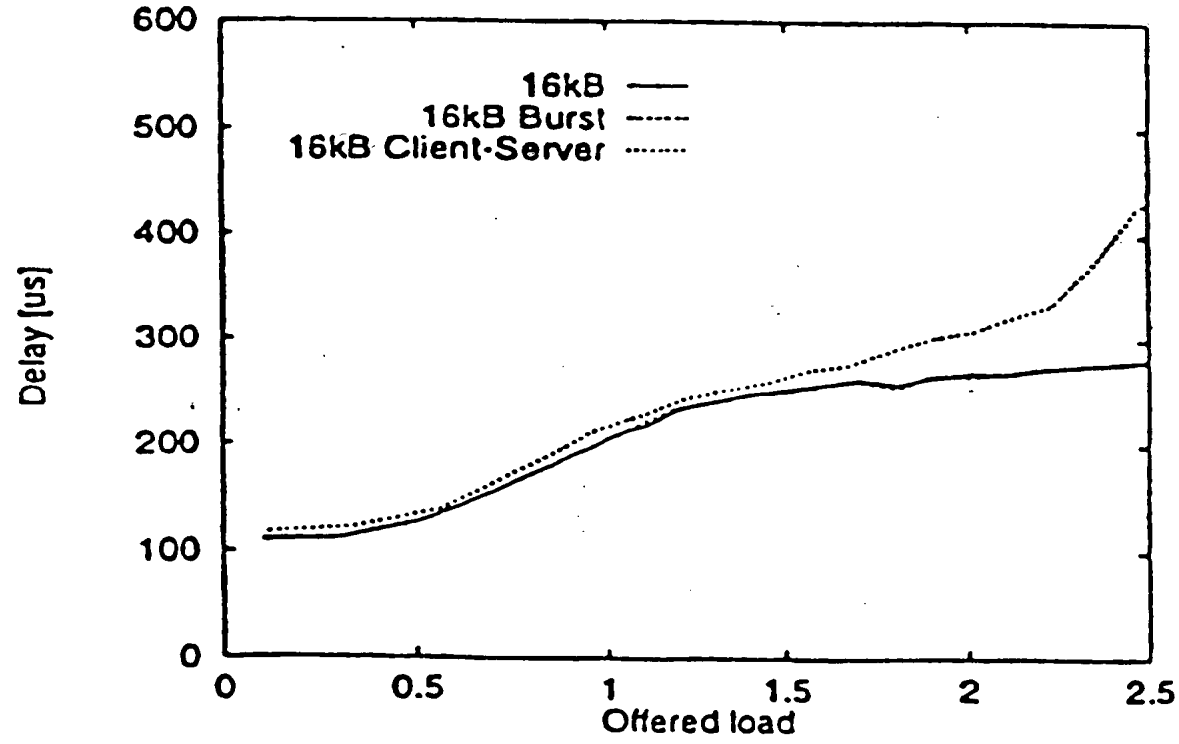

16kB
16kB Burst
16kB Client-Server
Delay [us]
Offered load
0
100
200
300
400
500
600
0.5
1
1.5
2
2.5

Fig. 23

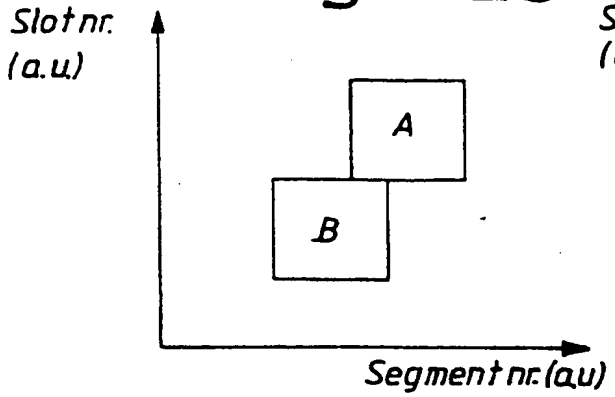


Fig. 24

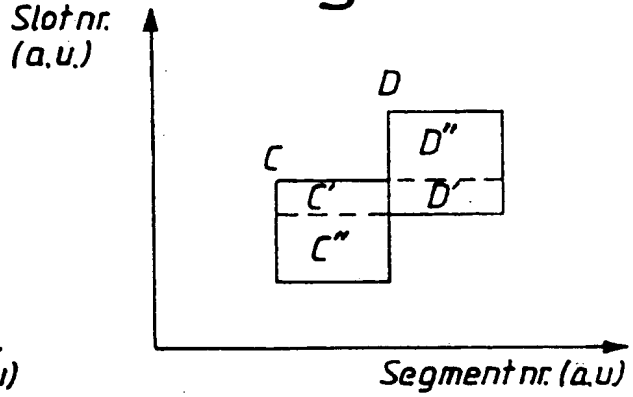


Fig. 25

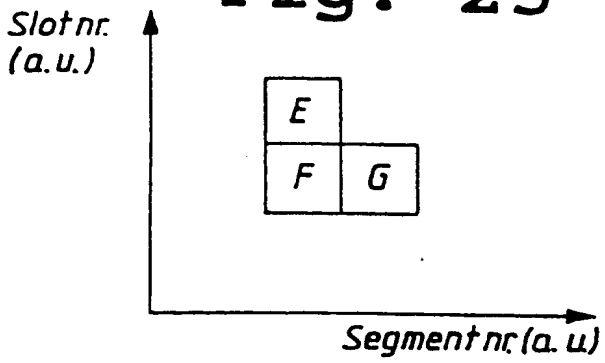


Fig. 26

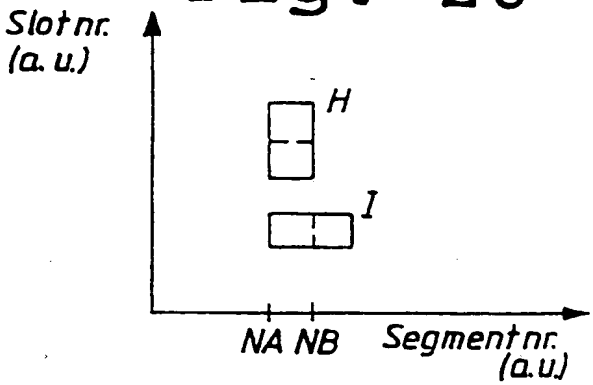


Fig. 27

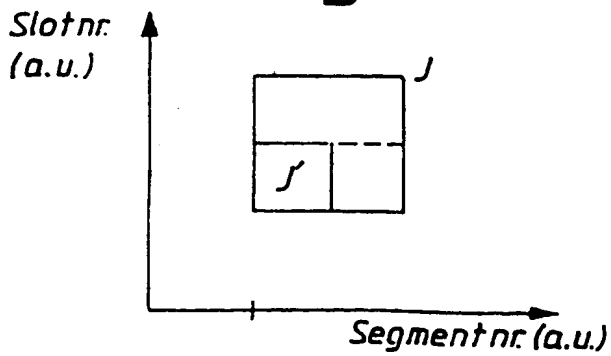
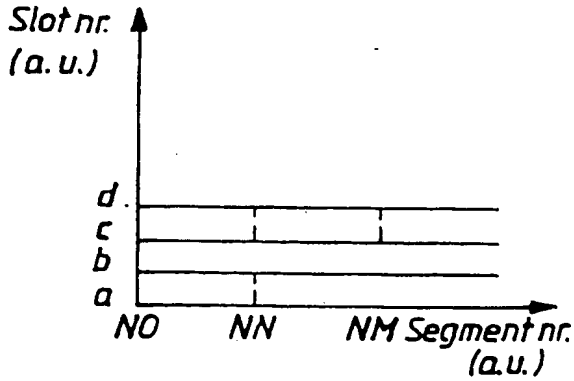


Fig. 28



INTERNATIONAL SEARCH REPORT

International application No.
PCT/SE 96/01750

A. CLASSIFICATION OF SUBJECT MATTER

IPC6: H04L 12/64, H04L 12/40, H04L 12/42
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC6: H04L, H04Q

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

SE,DK,FI,NO classes as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	Computer Networks and ISDN Systems, Volume 24, No 2, April 1992, (Amsterdam), Lars Gauffin, "Multi-gigabit networking based on DTM", page 119 - page 130, see especially page 122, 123, 126	6-9,19,20, 22,28-31, 35-37,40
Y		1,2,32,34
A		3-5,10-18, 21,23-27,33, 38,39

--

[X] Further documents are listed in the continuation of Box C. [X] See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" erlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or prior date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance: the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance: the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combinatic being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
27 March 1997	12.04.1997
Name and mailing address of the ISA/ Swedish Patent Office Box 5055, S-102 42 STOCKHOLM Facsimile No. +46 8 666 02 86	Authorized officer Anders Ströbeck Telephone No. +46 8 782 25 00

INTERNATIONAL SEARCH REPORT

International application No.
PCT/SE 96/01750

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	SE 460750 B (TELEFON AB L M ERICSSON), 13 November 1989 (13.11.89), page 4, line 1 - page 5, line 17	6-9,17,19, 20,22,28-31, 35-37
A		1-5,10-16, 18,21,23-27, 32-34,38-40
	--	
Y	EP 0451426 A1 (INTERNATIONAL BUSINESS MACHINES CORPORATION), 16 October 1991 (16.10.91), page 3, line 46 - page 4, line 4	1,2,32,34
A		3-31,33, 35-40
	--	
A	US 4553234 A (JOHAN R. BRANDSMA ET AL), 12 November 1985 (12.11.85), column 3, line 10 - line 60	1-40
	--	
P,X	IEEE Journal on Selected Areas in Comunications, Volume 14, No 2, February 1996, (USA), C. Bohm et al, "Fast Circuit Switching for the Next Generation of High Performance Networks" page 298 - page 305	1-23,28-40
P,A		24-27
	--	

INTERNATIONAL SEARCH REPORT

Information on patent family members

04/03/97

International application No.
PCT/SE 96/01750

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
SE 460750 B	13/11/89	SE 8800745 A	03/09/89
		WO 8908363 A	08/09/89
EP 0451426 A1	16/10/91	DE 69013886 D,T	18/05/95
		JP 2500080 B	29/05/96
		JP 4250737 A	07/09/92
		US 5173898 A	22/12/92
US 4553234 A	12/11/85	AU 562152 B	28/05/87
		AU 2309984 A	12/07/84
		CA 1228435 A	20/10/87
		EP 0115658 A,B	15/08/84
		SE 0115658 T3	
		JP 59135951 A	04/08/84
		NL 8300033 A	01/08/84

THIS PAGE BLANK (USPTO)

This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☒ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☒ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.

THIS PAGE BLANK (USPTO)